BAB II
TINJAUAN PUSTAKA

2.1 Karet

Karet (*Havea brasilliensis*) bukan tanaman asli dari Indonesia. Tanaman ini berasal dari Amerika bagian selatan tepatnya Brazil, maka dalam penulisan bahasa latinnya tertera asalnya yaitu *brasilliensis* (Amelia, 2016). Tanaman karet merupakan anggota famili *phorbiaceae*. Berbentuk pohon dengan tinggi 10-20 m, bercabang, dan mengandung banyak getah susu. Tanaman karet mengalami gugur daun sekali setahun pada musim kemarau. Dari tanaman karet, bagian yang paling banyak dimanfaatkan dan diolah untuk proses industri adalah getahnya yang biasa disebut lateks. Lateks dari tanaman karet dapat diperoleh melalui proses penyadapan. Penyadapan merupakan suatu tindakan pembukaan lateks, agar lateks yang terdapat di dalam tanaman karet keluar. Cara penyadapan yang telah dikenal luas adalah dengan mengiris sebagian dari kulit batang. Tanaman karet akan siap disadap untuk dapat diambil lateksnya tanpa menyebabkan gangguan yang berarti terhadap pertumbuhan dan kesehatannya apabila telah memenuhi kriteria sebagai berikut.

a. Keliling batang sudah mencapai 44-45 cm, di ukur pada ketinggian 150 cm dari tanah.
b. Umur tanaman 4-5 tahun.
c. Kondisi tanaman sehat.
d. Tinggi bukaan deresan 150 cm dari tanah.

2.2 Lateks

Lateks adalah suatu istilah yang dipakai untuk menyebutkan getah yang dikeluarkan dari pohon karet. Lateks terdapat pada bagian kulit, daun dan integumen biji karet. Lateks merupakan suatu larutan koloid dengan partikel karet dan bukan karet yang tersuspensi dalam suatu media yang mengandung beberapa macam zat. Lateks mengandung 25 – 40% bahan mentah dan 60 – 70%

serum yang terdiri dari air dan zat terlarut. Partikel karet tersuspensi (tersebar merata) dalam serum lateks dengan ukuran 0,04 – 3 mikron atau 0,2 milyar partikel padat permililiter lateks. Bentuk partikel bulat sampai lonjong. Berat jenis lateks 0,945 (pada 70^0F), serum 1,02 dan karet 0,91. Perbedaan berat jenis tersebut menyebabkan timbulnya cream pada permukaan lateks. Lateks membeku pada suhu 32^0F karena terjadi koagulasi.

Gambar 2.1 Getah Karet

2.3 Pengolahan Citra Digital

(Kusumanto & Tompunu, 2011) menjelaskan Pengolahan citra digital (*Digital Image Processing*) adalah sebuah disiplin ilmu yang mempelajari tentang teknik-teknik mengolah citra. Citra yang dimaksud disini adalah gambar diam (foto) maupun gambar bergerak (yang berasal dari webcam). Sedangkan digital disini mempunyai maksud bahwa pengolahan citra/gambar dilakukan secara digital menggunakan komputer.

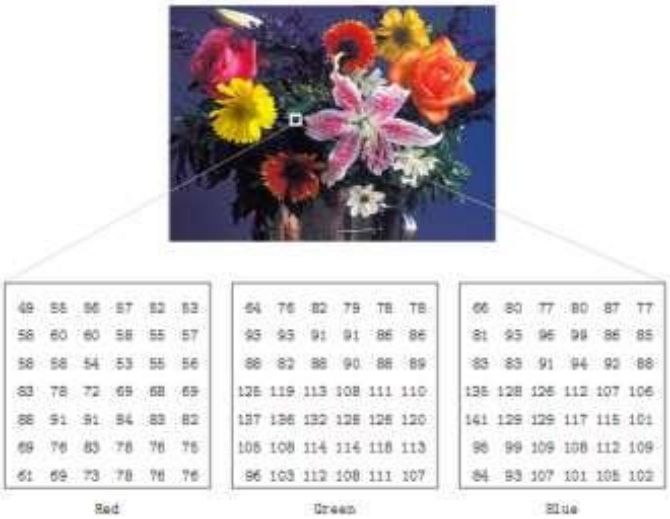
Secara matematis, citra merupakan fungsi kontinyu (*continue*) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Repersentasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi citra.

2.3.1 Jenis Citra Digital

Pada aplikasi pengolahan citra digital pada umumnya, citra digital dapat dibagi menjadi 3, *color image*, *balck and white image* dan *binary image*.

1. *Color Image* atau RGB (*Red, Green, Blue*).

   Pada color image ini masing-masing piksel memiliki warna tertentu, warna tersebut adalah merah (*Red*), hijau (*Green*) dan biru (*Blue*). Jika masing-masing warna memiliki range 0 - 255, maka totalnya adalah 2553 = 16.581.375 (16 K) variasi warna berbeda pada gambar, dimana variasi warna ini cukup untuk gambar apapun. Karena jumlah bit yang diperlukan untuk setiap pixel, gambar tersebut juga disebut gambar-bit warna. *Color image* ini terdiri dari tiga matriks yang mewakili nilai-nilai merah, hijau dan biru untuk setiap pikselnya, seperti yang ditunjukkan gambar

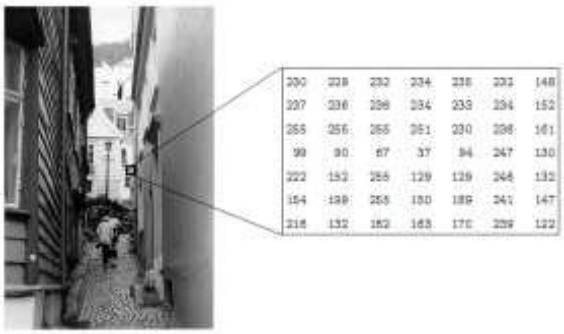


Gambar 2.2 *Color Image*

2. *Black and White*

   Citra digital *black and white* (*grayscale*) setiap pikselnya mempunyai warna gradasi mulai dari putih sampai hitam. Rentang tersebut berarti bahwa setiap piksel dapat diwakili oleh 8 bit, atau 1 byte. Rentang warna pada *black and white* sangat cocok digunakan untuk pengolahan file

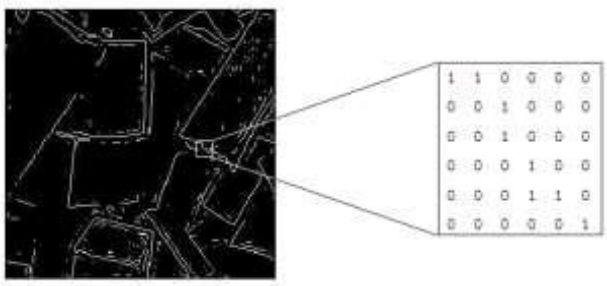
gambar. Salah satu bentuk fungsinya digunakan dalam kedokteran (X-ray).



Gambar 2.3 *Black and White (Grayscale)*

3. *Binary Image*

   Setiap piksel hanya terdiri dari warna hitam atau putih, karena hanya ada dua warna untuk setiap piksel, maka hanya perlu 1 bit per piksel (0 dan 1) atau apabila dalam 8 bit (0 dan 255), sehingga sangat efisien dalam hal penyimpanan. Gambar yang direpresentasikan dengan biner sangat cocok untuk teks (dicetak atau tulisan tangan), sidik jari (*finger print*), atau gambar arsitektur.



Gambar 2.4 *Binary Image*

2.4 Metode *K-Means Clustering*

K-Means adalah suatu metode penganalisaan data atau metode *Data Mining* yang melakukan proses pemodelan tanpa supervisi (*unsupervised*) dan merupakan salah satu metode yang melakukan pengelompokan data dengan sistem partisi. Metode *K-means* berusaha mengelompokkan data yang ada ke dalam beberapa kelompok, dimana data dalam satu kelompok mempunyai karakteristik yang sama satu sama lainnya dan mempunyai karakteristik yang

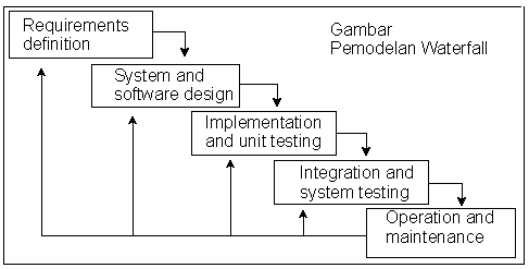
berbeda dengan data yang ada di dalam kelompok yang lain. Dengan kata lain, metode ini berusaha untuk meminimalkan variasi antar data yang ada di dalam suatu cluster dan memaksimalkan variasi dengan data yang ada di cluster lainnya.

Menurut (Nur, 2015) Algoritma *K-Means* secara umum dilakukan dengan algoritma dasar sebagai berikut:

a. Data yang ada dipisahkan dalam kelompok-kelompok data (klaster) k dan nilai-nilai data diacak ke dalam hasil-hasil kelompok data dalam kelompok data yang memiliki kesamaan jumlah dari nilai data.
b. Tiap nilai data dihitung menggunakan jarak *Euclidean* untuk tiap klaster.
c. Jika nilai data diwakili kelompok data tersendiri, biarkan, dan jika nilai data tak terwakili oleh kelompok data, pindah ke dalam kelompok data yang telah terwakili.
d. Ulangi langkah hingga lengkap meliputi seluruh hasil nilai data dalam perpindahan satu klaster ke klaster lainnya.

2.5 Model Waterfall

Model ini melakukan pendekatan secara sistematis dan urut mulai dari level kebutuhan sistem lalu menuju ke tahap analisis, desain, *coding, testing / verification, dan maintenance.* Disebut dengan *waterfall* karena tahap demi tahap yang dilalui harus menunggu selesainya tahap sebelumnya dan berjalan berurutan. Sebagai contoh tahap desain harus menunggu selesainya tahap sebelumnya yaitu tahap *requirement* (Pascapraharastyan, Rizki Alfiasca., Supriyanto, Antok., Sudarmaningtyas, 2014). Secara umum tahapan pada model *waterfall* dapat dilihat pada gambar berikut.



Gambar 2.5 Siklus Pengembangan Dengan Metode *Waterfall*

1. ***Requirements definition***

   Proses pencarian kebutuhan diintensifkan dan difokuskan pada *software*. Untuk mengetahui sifat dari program yang akan dibuat, maka para *software engineer* harus mengerti tentang domain informasi dari *software*, misalnya fungsi yang dibutuhkan, *user interface*. Dari 2 aktivitas tersebut (pencarian kebutuhan sistem dan *software*) harus didokumentasikan dan ditunjukkan kepada pelanggan.

2. ***System and Software Design***

   Proses ini digunakan untuk mengubah kebutuhan- kebutuhan diatas menjadi representasi ke dalam bentuk *"blueprint" software* sebelum *coding* dimulai. Desain harus dapat mengimplementasikan kebutuhan yang telah disebutkan pada tahap sebelumnya. Seperti 2 aktivitas sebelumnya, maka proses ini juga harus didokumentasikan sebagai konfigurasi dari *software.*

3. ***Implementation and Unit Testing***

   Untuk dapat dimengerti oleh mesin, dalam hal ini adalah komputer, maka desain tadi harus diubah bentuknya menjadi bentuk yang dapat dimengerti oleh mesin, yaitu ke dalam bahasa pemrograman melalui proses *coding*. Tahap ini merupakan implementasi dari tahap design yang secara teknis nantinya dikerjakan oleh *programmer*.

4. ***Integration and System Testing***

   Sesuatu yang dibuat haruslah diujicobakan. Demikian juga dengan *software*. Semua fungsi-fungsi *software* harus diujicobakan, agar

software bebas dari *error*, dan hasilnya harus benar-benar sesuai dengan kebutuhan yang sudah didefinisikan sebelumnya.

5. ***Operation and Maintenance***

   Pemeliharaan suatu *software* diperlukan, termasuk di dalamnya adalah pengembangan, karena *software* yang dibuat tidak selamanya hanya seperti itu. Ketika dijalankan mungkin saja masih ada error kecil yang tidak ditemukan sebelumnya, atau ada penambahan fitur-fitur yang belum ada pada *software* tersebut. Pengembangan diperlukan ketika adanya perubahan dari *eksternal* perusahaan seperti ketika ada pergantian sistem operasi, atau perangkat lainnya.

2.6 UML (*Unified Markup Languange*)

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah system (Dharwiyanti & Wahono, 2003). Alat bantu yang dipergunakan dalam perancangan berorientasi objek berbasis UML adalah sebagai berikut.

2.6.1 *Use Case Diagram*

Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem yang dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih actor dengan sistem yang akan dibuat. Dapat dikatakan use case digunakan untuk mengetahui fungsi apa saja yang dibuat dan siapa saja yang berhak mengunakan fungsi-fungsi tersebut. Berikut simbol yang digunakan pada *use case diagram* :

Tabel 2.1 Simbol *Use Case Diagram*

Gambar	Keterangan
	Use case menggambarkan fungsionalitas yang disediakan system sebagai unit-unit yang bertukar pesan antar unit dengan actor, biasanya dinyatakan dengan mengunakan kata kerja diawal nama use case.
Actor	Aktor adalah orang lain atau system lain yang mengaktifkan fungsi dari target system. Untuk megidentifikasikan actor, harus ditentukan pembagian kerja dan tugas-tugas yang berkaitan dengan peran pada target system. Orang atau system bias muncul dalam beberapa peran.
	Ososiasi antara actor dan use case yang mengunakan panah terbuka untuk mengindikasikan bila actor berinteraksi secara pasif dengan system.
	Ososiasi antara actor dan use case yang mengunakan garis untuk mengindikasikan siapa atau apa yang meminta interaksi langsung bukan mengindikasikan aliran data.
<<include>>	Include merupakan pemangilan use case oleh use case lainya , contohnya pemanggilan sebuah fungsi program.
<<extend>>	Perluasan dari use case lain jika kondisi atau syarat terpenuhi.

2.6.2 *Activity Diagram*

Secara grafis digunakan untuk menggambarkan rangkaian aliran aktivitas baik proses bisnis maupun *use case*. *Activity diagram* dapat juga digunakan untuk memodelkan action yang akan dilakukan saat sebuah operasi dieksekusi, dan

memodelkan hasil dari action tersebut**.** Simbol- simbol yang digunakan dalam *activity diagram* yaitu :

Tabel 2.2 Simbol *Activity Diagram*

Gambar	**Keterangan**
Activity	Mengambarkan suatu proses/kegiatan sistem
State	Menunjukan eksekusi dari suatu aksi
	Start Point merupakan awak dari suatu aktivitas
	End point, akhir aktivitas
	Fork/Rake Node Satu aliran atau beberapa aliran yang pada tahap tertentu berubah menjadi satu atau beberapa aliran.
Swimlane	Swinlane, pembagian activity diagram untuk menunjukan siapa melakukan apa

2.6.3 *Class Diagram*

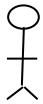
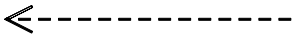
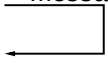
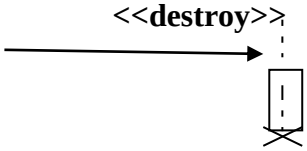
Class diagram merupapakan hubungan antara kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sitem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class diagram* juga menunjukan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan.

Class diagram secara khas meliputi : kelas (*class*), relasi, *Associations*, *Generalizion* dan *Agregation*, Atribut, operasi, dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau object.

2.6.4 *Sequence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeksripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel 2.3 Simbol *Sequence Diagram*

Gambar	Keterangan
	Orang, proses atau system lain yang berinteraksi dengan sistem
1: Masukan	Menyatakan bahwa suatu objek mengirimkan data atau masukan atau informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
	Activation, activation mewakili sebuah eksekusi operasi dari objek.
1: Keluaran 	Message return, simbol membalas pesan
1: [condition] message name 	Recursive, mengambarkan pengiriman pesan untuk dirinya sendiri.
<<create>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah objek yang di buat.
<<destroy>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri,

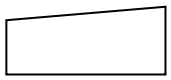
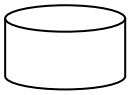
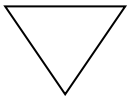
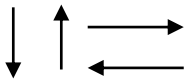
	sebaiknya jika ada create maka ada destroy.
Garis Hidup / Lifeline	Menyatakan kehidupan suatu objek

2.7 Bagan Alir Program (*Flowchart*)

Bagan alir (*flowchart*) adalah bagan (*chart*) yang menunjukan alir (*flow*) didalam program atau prosedur sistem secara logika Bagan alir digunakan terutama untuk alat bantu komunukasi dan untuk dokumentasi (Permatasari & Ardy, 1978). Simbol-simbol yang digunakan dalam bagan alir program dapat dilihat pada tabel 2.4.

Tabel 2.4 Simbol-simbol bagan alir program *(Flowchart)*

Simbol	Keterangan
	Terminator Menunjukkan awal dan akhir suatu proses.
	Input / Output Menunjukkan data masukan (*input*) atau data keluaran (*output*).
	Prosedur Menunjukkan proses data.
	Process Menunjukkan kegiatan proses dari operasi program computer
	Decision Menunjukkan penjelasan suatu proses

	*Keybord* Memasukkan atau input data melalui keyboard
	*Harddisk* Tempat penyimpanan data pada komputer
	Simpan *Offine* Penyimpanan arsip manual
	Connector Menunjukkan penghubung ke halaman yang sama
	Off-page Connentor Menunjukkan penghubung ke halaman yang berbeda
	Garis Alir Menunjukkan simbol dari alur system atau proses

2.8 Black Box Testing

Black Box Testing berfokus pada spesifikasi fungsional dari perangkat lunak. Tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program (Mustaqbal, 2015).

Black Box Testing cenderung untuk menemukan hal-hal berikut:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka (*interface errors*).
3. Kesalahan performansi (*performance errors*).
4. Kesalahan inisialisasi dan terminasi.