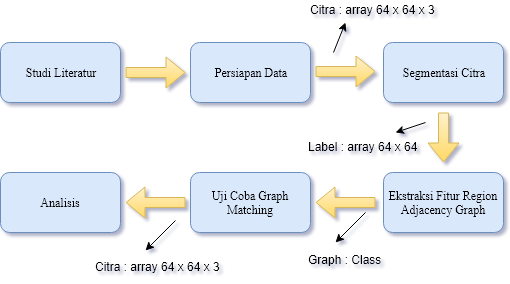
# BAB III

# METODOLOGI PENELITIAN

Bab ini berisi uraian mengenai desain sistematika penelitian pengenalan objek citra digital menggunakan ekstraksi *region adjacency graph* dan *graph matching*. Pada bab ini juga akan diuraikan mengenai alat dan bahan yang digunakan selama proses penelitian.

## Desain Sistematika Penelitian

Penelitian ini dilakukan untuk membangun sebuah program analisis berbasis *graph* yang dapat digunakan untuk mengenali citra digital. Seluruh proses analisis mulai dari segmentasi, ekstraksi fitur, dan pengklasifikasian akan dilakukan dengan metode berbasis *graph*. Data yang digunakan berupa citra digital sederhana yang merepresentasikan sebuah objek. Adapun untuk proses pengujian, akan dilakukan dengan menguji setiap data terhadap keseluruhan dataset. Gambar 3.1 menunjukkan metode penelitian yang akan dilakukan.



Gambar 3.1 Metode Penelitian

### Studi Literatur

Pada tahap ini dilakukan studi literatur terhadap penelitian-penelitian yang terkait dengan operasi pengenalan citra berbasis *graph*, mulai dari segmentasi, ekstraksi fitur, dan pengklasifikasian. Pada tahap ini dipelajari teknik-teknik yang telah menjadi *state-of-the-art* sehingga dapat menjadi rujukan dalam pembangunan metode penelitian.

Pada tahap ini juga dipelajari bagaimana penentuan spesifikasi citra digital agar memenuhi dalam penelitian ini, sehingga diharapkan dapat bekerja optimal dalam proses pengenalan citra pada metode yang dilakukan.

### Persiapan Data

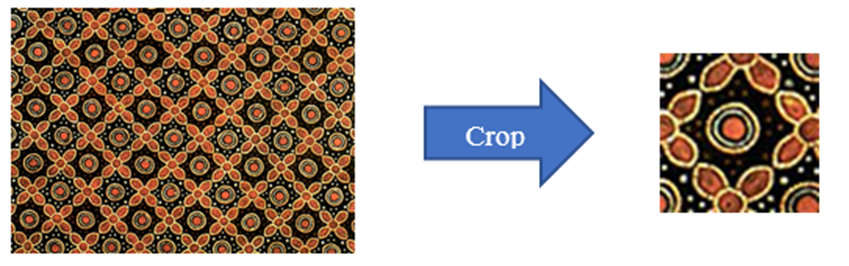
Pada tahap ini akan dilakukan persiapan data sesuai dengan batasan masalah yang telah ditentukan sebelumnya. Data sintetis yang dibuat dalam proses penelitian ini sebanyak 200 buah citra digital dengan ukuran 64 x 64 piksel. Data terdiri dari 4 kelas berbeda sehingga masing-masingnya berjumlah 50 buah. Sampel data citra ditunjukkan pada Gambar 3.2 Sampel Dataset Citra Sintetis.



Gambar 3.2 Sampel Dataset Citra Sintetis

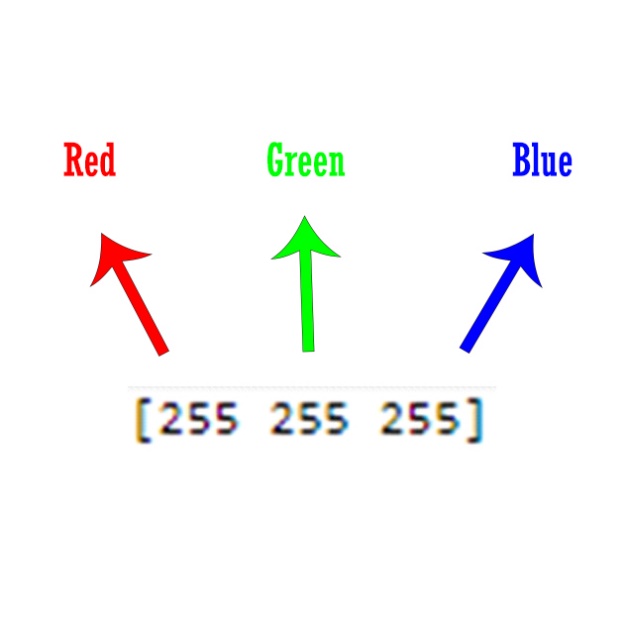
Data tersebut dibuat menggunakan Adobe Photoshop CS6 64 bit. Citra digambar secara manual pada halaman 64 x 64 dengan menambahkan sejumlah *shape* dengan warna yang telah ditentukan. Kemudian masing-masing citra disimpan dalam ekstensi berkas \*.jpg di dalam media penyimpanan.

Untuk dataset *real* yang digunakan adalah motif inti batik. Motif inti batik yang digunakan pada penelitian ini terdiri dari 3 varian: 1) Batik Grompol, 2) Batik Parang, dan 3) Batik Kawung. Setiap jenis motif batik terdiri masing-masing 60 citra untuk dimasukkan ke dalam data set. Citra dipersiapkan melalui *adobe photoshop* dengan teknik *cropping* dari citra batik utuh. Gambar 3.3 menunjukkan contoh hasil pengambilan motif inti batik dari citra batik utuh.



Gambar 3.3 Contoh hasil pengambilan motif inti batik

Dalam bahasa pemrograman *python* penginputan citra dibantu dengan modul *io* dari *library scikit-image*, kemudian setiap citra direpresentasikan dalam matriks 3 dimensi dengan nilai 64 x 64 x 3. Dimana nilai 64 pertama menunjukkan baris piksel, nilai 64 kedua kolom piksel, dan nilai 3 menunjukkan jumlah *channel* pada citra. Visualisasi dari representasi tiap piksel pada bahasa pemrograman *python* ditunjukkan oleh Gambar 3.4 Representasi Satu Piksel dalam Python



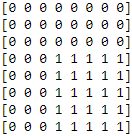
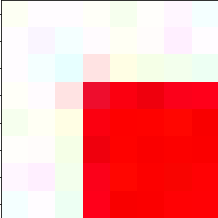
Gambar 3.4 Representasi Satu Piksel dalam Python

### Segmentasi Citra

Pada tahap ini akan dilakukan proses segmentasi citra menjadi bagian-bagian yang lebih sederhana. Metode yang digunakan pada penelitian ini merupakan segmentasi melalui pendekatan berbasis *graph* dengan metode yang diusulkan oleh Felzenzswalb.

Proses segmentasi ini dibantu dengan *library* dari *scikit-image* melalui sebuah fungsi ***segmentation.felzenzswalb***. Kemudian dilakukan penyetelanterhadap parameter dari fungsi tersebut, untuk mendapatkan hasil segmentasi sesuai dengan persepsi jumlah *region* yang seharusnya.

*Input* pada tahap ini adalah citra digital 64 x 64 x 3. Setelah diproses, *output* dihasilkan berupa label dengan tipe data *array* n-dimensi ukuran *64* x 64 yang menunjukkan label *region* untuk setiap piksel. Gambar 3.5 menunjukkan contoh sederhana segmentasi pada citra 8 x 8 x 3, sehingga didapatkan label berupa array 8 x 8. Selanjutnya label akan disimpan dalam format \*.csv di dalam media penyimpanan.

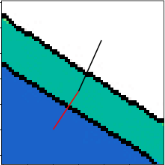
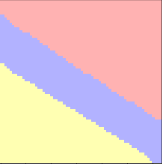


Segmentasi

Gambar 3.5 Representai hasil segmentasi

### Ekstraksi Fitur melalui Pembangunan RAG

Di tahap ini akan dilakukan ekstraksi fitur pada setiap citra yang telah tersegmentasi melalui pembangunan *Region Adjacency Graph* (RAG). RAG dibangun dengan menghitung nilai *centroid* dari setiap label segmentasi, menghitung nilai rata-rata RGB pada setiap *centroid*, kemudian menghubungkan antar *centroid* menggunakan *eucledian distance*. Gambar 3.6 merupakan penggambaran proses ekstraksi RAG.



Segmentasi

RAG

Gambar 3.6 Penggambaran proses ekstraksi RAG

*Output* dari tahap ini akan menghasilkan sebuah *graph* untuk masing-masing citra yang berperan sebagai hasil ekstraksi fitur dalam penelitian ini. Fitur-fitur yang dihasilkan dari setiap *graph* berupa:

* + - 1. *Nodes*, simpul-simpul yang terbentuk
      2. *Total Color*, total nilai piksel RGB dari masing-masing simpul
      3. *Centroid*, nilai titik tengah dari masing-masing simpul
      4. *Mean Color*, nilai rata-rata piksel dalam RGB dari masing-masing simpul
      5. *Edges*, sisi-sisi yang terbentuk dari setiap simpul
      6. *Weight*, nilai *eucledian distance* dari masing-masing *edge*

Melalui bantuan *library* *Networkx* pada bahasa pemrograman *python*, *graph* yang dihasilkan akan memiliki tipe data berupa *class* dari *library* tersebut. Kemudian *graph* dari RAG masing-masing citra akan disimpan pada media penyimpanan dalam format berkas \*.gpickle.

### *Graph Matching*

Pada langkah ini akan diberikan citra sebagai masukan untuk memperoleh data yang memiliki kemiripan dengan citra tersebut berdasarkan algoritma *graph matching* yang ditetapkan. Adapun algoritma yang digunakan yaitu algoritma *VF2* dan *Graph Edit Distance*. *Output* dari langkah ini berupa citra digital yang memiliki kemiripan dengan citra uji yang diberikan dalam setiap percobaan.

Citra yang akan diujikan diambil dari dataset, dan akan diproses terhadap setiap dataset. Citra tersebut akan diberikan perlakuan yang sama seperti pada dataset yaitu segmentasi, ekstraksi fitur RAG, kemudian dilakukan pencocokkan terhadap keseluruhan dataset.

Algoritma *VF2* bekerja berdasarkan prinsip *graph isomorphism*, sehingga pada algoritma ini tidak dibutuhkan parameter tertentu sebagai nilai masukan dalam penelitian ini. Sedangkan untuk *Graph Edit Distance* dapat dilakukan penyetelan parameter berupa batasan nilai *cost* untuk setiap hasil pencocokkan, sehingga hasilnya dapat dibuat lebih fleksibel.

## Alat dan Bahan Penelitian

Dalam penelitian untuk mengenali citra melalui pendekatan berbasis *graph* ini dibutuhkan perangkat keras dan perangkat lunak sebagai berikut.

1. Perangkat Keras

Perangkat keras yang digunakan adalah sebuah komputer yang memiliki spesifikasi sebagai berikut:

* *Processor* : Intel® Celeron® CPU B820 @ 1.70GHz
* RAM : 4.00 GB
* *Disk Space* : HDD 500 GB 5400 rpm
* *Video Graphics* : Intel HD Graphic

1. Perangkat Lunak

Perangkat lunak yang digunakan adalah sebagai berikut:

* Sistem Operasi : Windows 8.1 Enterprise 64-bit
* Bahasa Pemrograman : Python 2.7
* Aplikasi : Sublime Text 3, Adobe Photoshop CS6, Paint
* Library Tambahan : Numpy, Scikit-Image, Matplotlib, Networkx 1.9, Scipy, Venv, Flask, Jupyter notebook

## Perancangan Program Aplikasi Antarmuka

Perancangan program aplikasi antarmuka sederhana ini dilakukan guna memudahkan *end-user* dalam menggunakan program hasil analisis dari penelitian ini. Dalam program antarmuka yang dibangun ini akan memuat program yang dapat mempermudah dalam menampilkan hasil segmentasi dan hasil pencocokkan citra-citra yang dianggap memiliki kemiripan untuk satu masukan citra.

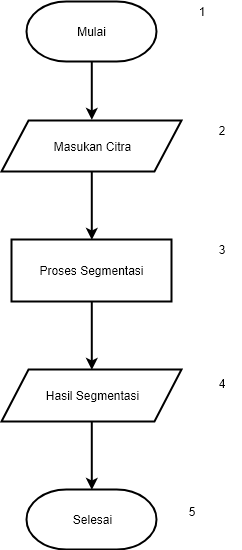
Citra yang dimasukan akan diproses dengan parameter statis, baik pada tahap segmentasi maupun *graph* matching. Parameter tersebut merupakan hasil analisis dari program konsep dalam *jupyter notebook*. Sehingga parameter yang dapat diubah hanya citra masukannya saja.

### *Flowchart* Program

Pada program aplikasi antarmuka ini akan dibangun dua program utama, yaitu program untuk menampilkan segmentasi pada citra masukan dan program untuk pemrosesan graph matching serta menampilkan hasilnya. Program-program berikut akan dibangun menggunakan bahasa pemrograman python, dan menggunakan html sebagai media antarmuka.

1. *Flowchart* Segmentasi Citra

Berikut cara kerja program untuk menampilkan segmentasi pada citra:

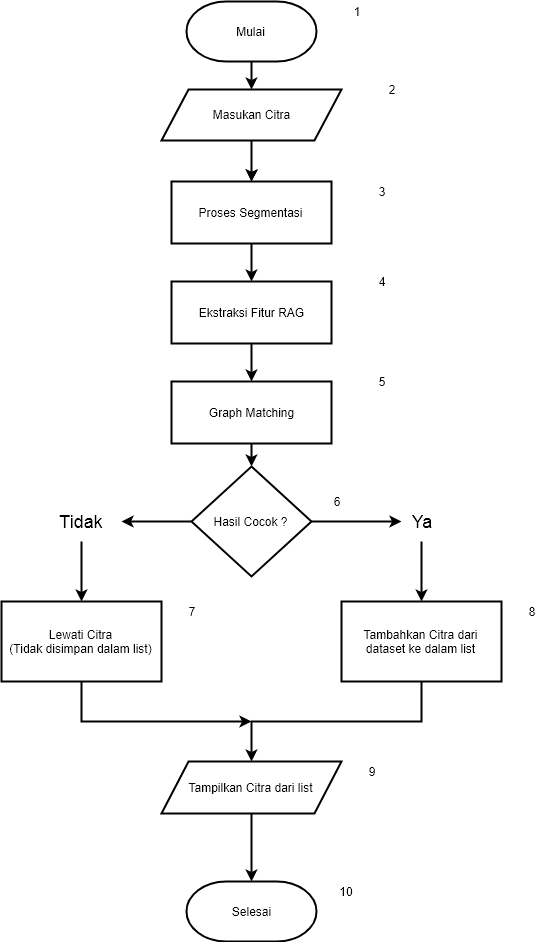


Gambar 3.7 Flowchart Segmentasi Citra

Seperti yang ditunjukkan pada Gambar 3.7 dijelaskan *flowchart* program untuk menampilkan segmentasi citra serta mencetak nilai-nilai label citra hasil segmentasi. Berikut adalah penjelasan *flowchart* segmentasi citra:

1. Merupakan *state* awal dari aplikasi. Pada *state* ini dilakukan proses pengaktifan halaman *web* melalui *localhost*
2. Pengguna memilih citra yang akan di proses dari folder yang telah ditentukan
3. Segmentasi di dalam program akan menghasilkan nilai-nilai label berukuran *array* 64 x 64 dari citra masukan
4. Hasil segmentasi beserta nilai-nilai label akan ditampilkan pada halaman antarmuka
5. *State* akhir aplikasi
6. *Flowchart* Pencocokan Citra

Berikut cara kerja program untuk pencocokkan citra serta menampilkan hasilnya dalam halaman antarmuka:



Gambar 3.8 Flowchart Pencocokkan Citra

Pada Gambar 3.8 dijelaskan *flowchart* pencocokkan citra masukan dengan seluruh citra dalam dataset.

Berikut adalah penjelasan *flowchart* pencocokkan citra:

1. Merupakan *state* awal dari aplikasi. Pada *state* ini dilakukan proses pengaktifan halaman *web* melalui *localhost*
2. Pengguna memilih citra yang akan di proses dari folder yang telah ditentukan
3. Segmentasi di dalam program akan menghasilkan nilai-nilai label berukuran *array* 64 x 64 dari citra masukan
4. Ekstraksi fitur akan membangun sebuah *Region Adjacency Graph* (RAG) dari citra dan label yang telah dihasilkan dari proses segmentasi sebelumnya. RAG yang dihasilkan akan disimpan dalam bentuk *class* yang sesuai untuk *library* networkx pada bahasa pemrograman python
5. Proses ini akan melakukan pencocokkan RAG dari citra masukan dengan setiap RAG dalam dataset. Metodeyang akan digunakan adalah *graph matching* melalui pendekatan *Graph Edit Distance* (GED) dengan parameter batasan yang statis
6. Program melakukan pengecekan hasil pencocokkan. Jika citra dianggap cocok, program akan dilanjutkan ke *state* 8. Jika tidak cocok, program akan dilanjutkan ke *state* 7.
7. Citra yang dicocokkan akan dilewati dan tidak disimpan dalam *list*
8. Citra yang dicocokkan akan disimpan dalam *list*
9. Menampilkan citra-citra dari *list* ke dalam halaman *web*.

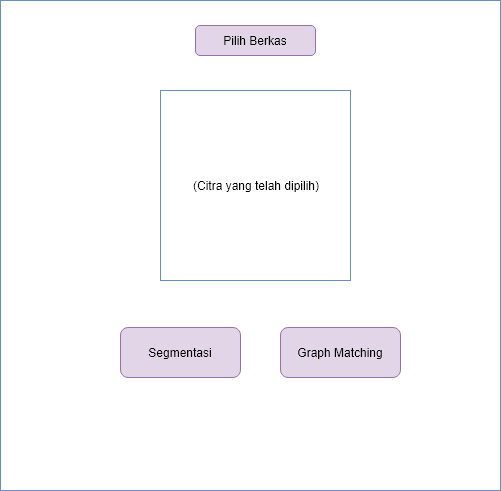
### Perancangan Desain Antarmuka Program

Desain antarmuka pada program ini akan dikembangkan dalam bentuk halaman *web*. *Server* yang digunakan merupakan *localhost* yang disiapkan dari modul *flask* dari bahasa pemrograman python.

Berikut adalah rancangan desain antarmuka dari program aplikasi ini:

1. Halaman Masukan Citra

Halaman masukan adalah halaman awal pada saat program aplikasi dijalankan. Halaman ini ditujukan untuk memilih berkas citra dari folder yang telah ditentukan.

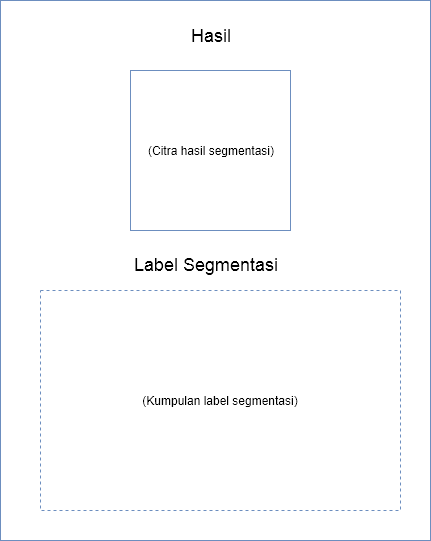


Gambar 3.9 Halaman Masukan Citra

Desain antarmuka halaman masukan pada Gambar 3.9 yang dirancang memiliki:

1. *Button* “pilih berkas” berfungsi untuk memilih berkas citra melalui *file explorer* dari sistem operasi
2. *Image “citra yang dipilih”* berfungsi untuk menampilkan citra yang telah dipilih dari *button* “pilih berkas
3. *Button* “segmentasi” berfungsi untuk memproses serta menampilkan segmentasi dari citra terpilih
4. *Button* “*graph matching*” berfungsi untuk melakukan dan menampilkan hasil dari proses pencocokkan secara keseluruhan pada citra terpilih dengan setiap dataset yang telah disediakan.
5. Halaman Segmentasi Citra

Halaman segmentasi adalah halaman untuk menampilkan hasil segmentasi pada citra secara visual, dan juga mencetak nilai-nilai label dari hasil segmentasi tersebut.

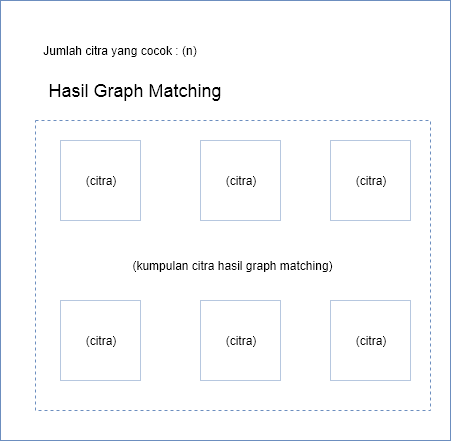


Gambar 3.10 Halaman Segmentasi Citra

Desain antarmuka halaman hasil segmentasi pada Gambar 3.10 yang dirancang memiliki:

1. *Text* “hasil” berfungsi sebagai judul untuk konten citra hasil segmentasi di bawahnya
2. *Image* “citra hasil segmentasi” berfungsi untuk menampilkan citra hasil segmentasi
3. *Text “*label segmentasi” berfungsi sebagai judul untuk konten label citra hasil segmentasi yang berada di bawahnya
4. *Text* “kumpulan label segmentasi” berfungsi untuk mencetak nilai-nilai label hasil segmentasi citra sebanyak 64 x 64
5. Halaman *Graph Matching*

Halaman *graph matching* adalah halaman untuk menampilkan citra hasil proses pencocokan citra masukan dengan setiap pada dataset secara visual, serta mencetak jumlahnya.



Gambar 3.11 Halaman Graph Matching

Desain antarmuka halaman hasil *graph* matching pada Gambar 3.10 yang dirancang memiliki:

1. *Text* “jumlah citra cocok: n” berfungsi untuk menunjukkan jumlah citra yang tampil atau dianggap cocok dengan citra masukan
2. *Text* “hasil *graph matching*” berfungsi sebagai judul untuk citra hasil *graph matching* yang berada di bawahnya
3. *Image “*citra” merupakan citra-citra yang ditampilkan sebagai hasil yang dianggap cocok dengan citra masukan dari proses *graph matching*