BAB III. METODOLOGI PENELITIAN

3.1 Data

Data yang digunakan merupakan gambar hasil tangkapan kamera. Contoh data berupa citra hasil tangkapan kamera *smartphone*

Gambar 3. 1 Contoh citra yang mengandung teks.

3.2 Akuisisi Data

Data yang diperoleh adalah teks yang ada di sekitar. Untuk pengambilan teks tersebut menggunakan kamera *smartphone*, mengambil foto dari kamera *smartphone*

3.3 Metode Pengolahan Data

a. *File Input*

File input berupa tangkapan citra digital dengan format *Bitmap* (**.BMP)*. Penulis menggunakan *Bitmap* karena *Bitmap* merupakan file tidak terkompresi, sedangkan JPEG adalah file terkompresi. Format PNG hampir serupa dengan format BMP, namun PNG mempunyai proses *compress*, yang menjadikan *file* nya mempunyai ukuran yang lebih kecil.

b. *Crop*

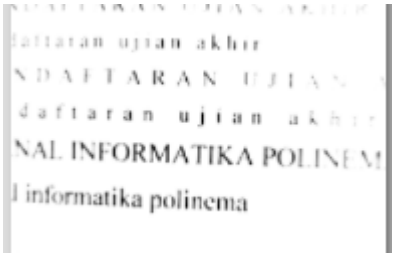
Crop atau disebut dengan pemotongan adalah proses pemotongan gambar untuk mendapatkan gambar yang terdapat objek yang ingin diolah dengan lingkup kecil dari gambar asalnya. Pemotongan citra dilakukan untuk mempermudah pemrosesan huruf. Proses ini dibantu dengan *library* Google

Vision untuk mencari posisi huruf pada citra dan berapa banyak huruf yang dideteksi. Sistem akan mendapatkan sebuah nilai *RectF*. *RectF* memegang 4 koordinat dengan satuan *float* untuk persegi panjang. Persegi panjang diwakili oleh koordinat 4 tepinya (kiri, atas, kanan, bawah).

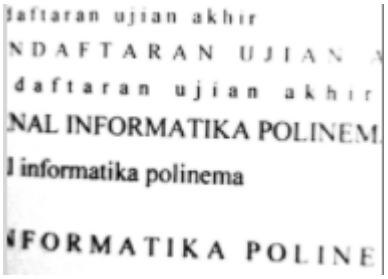
Setelah mendapatkan 4 koordinat, dilakukan perhitungan jumlah huruf untuk satu persegi tersebut. Selanjutnya mencari panjang dari persegi. Setelah menemukan panjang, dilakukan pemotongan per huruf. Langkah selanjutnya yaitu memindahkan huruf hasil pemotongan tersebut ke bentuk citra baru, dibuatkan sebuah *canvas*. Untuk luas *canvas* nya didapatkan dari perhitungan mencari panjang dan lebar dari 4 koordinat *pixel* sebelumnya. Langkah ini dilakukan agar citra asli tidak hilang / tidak tertindih dengan citra yang lain, yang menyebabkan hilangnya citra masukkan / citra hasil tangkapan kamera.

c. *Thresholding*

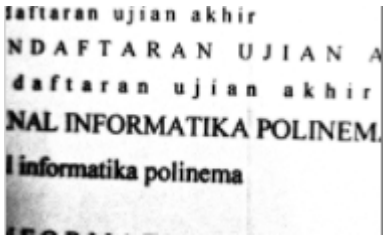
Thresholding yaitu proses untuk memisahkan antara objek dengan *background* pada *image*. Inputan citra pada *thresholding* ini bisa berupa *grayscale image* atau *color image*. *Output* dari proses *thresholding* ini berupa *binary image*. *Pixel binary image* hanya memiliki 2 nilai intensitas yaitu 0 untuk *pixel* berwarna hitam, dan 1 untuk *pixel* berwarna putih. Sedangkan parameter yang digunakan pada proses *thresholding* yaitu *intensity threshold*. Jika nilai intensitas *pixel* lebih tinggi daripada *intensity threshold* maka *pixel* diubah menjadi warna putih, sebaliknya jika nilai intensitas *pixel* lebih rendah maka *pixel* diubah menjadi warna hitam. Untuk penelitian ini penulis menggunakan nilai *threshold* 100, karena nilai tersebut sudah pas untuk tulisan berwarna hitam. Jika kurang dari 100, hasil citra menunjukkan beberapa huruf mengalami kabur / tidak jelas. Jika lebih dari 100, hasil citra menunjukkan banyak *noise* warna hitam di luar area huruf



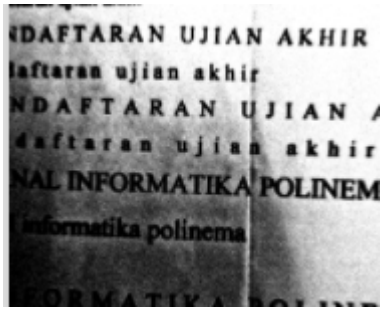
Gambar 3. 2 Hasil citra dengan nilai *thresholding* 75



Gambar 3. 3 Hasil citra dengan nilai *thresholding* 100



Gambar 3. 4 Hasil citra dengan nilai *thresholding* 125



Gambar 3. 5 Hasil citra dengan nilai *thresholding* 150

e. Mengubah ukuran citra

Langkah ini dilakukan untuk menyamakan ukuran citra yang diproses dengan ukuran citra *template*. Langkah pertama untuk mengubah ukuran citra yaitu mencari skala citra, menggunakan rumus 3.1 berikut:

$$scale = \frac{newSize}{src(height)} \tag{3.1}$$

Keterangan:

scale = skala citra

newSize = nilai ukuran baru (pixel)
src(height) = nilai tinggi dari citra yang dimasukkan (pixel)

Kemudian menentukan nilai lebar dan tinggi yang baru, menggunakan rumus 4.6 dan 4.7 berikut:

$$newWidth = src.width\ x\ scale \quad (3.2)$$

$$newHeight = scale \quad (3.3)$$

Keterangan:
newWidth = nilai lebar baru (pixel)
newHeight = nilai tinggi baru (pixel)
src.width = nilai lebar pada citra masukkan (pixel)

f. Template Matching

Pada langkah ini dilakukan pencocokkan citra hasil pengolahan sebelumnya dengan semua citra *template* yang ada di sistem. Jumlah *template image* yang ada pada aplikasi ini berjumlah 62 citra, yaitu 26 citra huruf kecil, 26 citra huruf besar, dan 10 citra angka.

Gambar 3. 6 Contoh citra *template* huruf "a" kapital

Gambar 3. 7 Contoh citra *template* huruf "m"

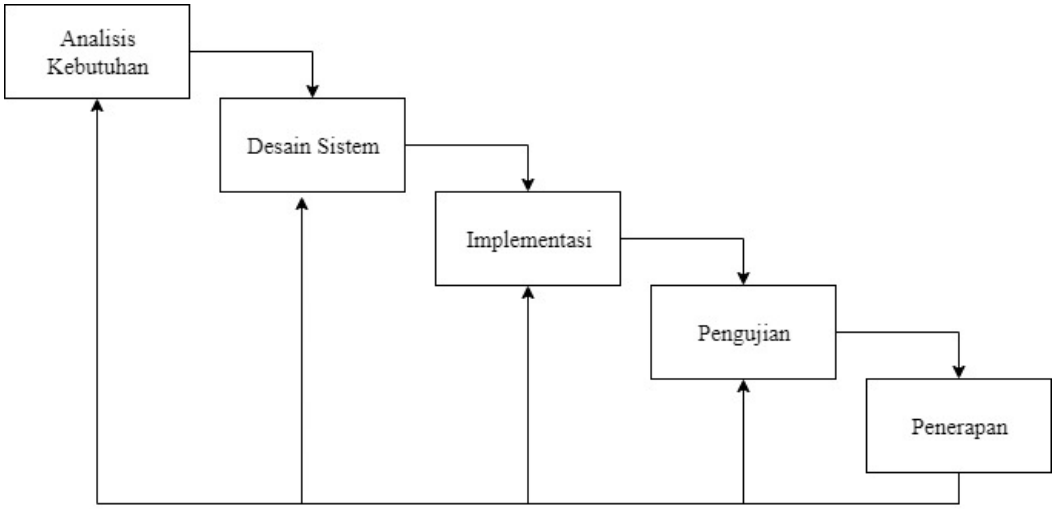
Gambar 3. 8 Contoh citra *template* angka “4”

g. Text to Speech

Untuk mengubah / mengonversi teks menjadi suara, pada penelitian ini menggunakan TTS API (*Text-To-Speech API)* di Android. Mesin TTS ini mendukung sejumlah Bahasa, seperti Inggris, Prancis, Jerman, Italia, dan Spanyol. Karena berada di Indonesia, mesin TTS di atur menjadi Bahasa Indonesia. Cara kerja mesin ini sistem harus mempunya *text* / kalimat yang akan diolah. Kemudian sistem dapat memanggil mesin TTS untuk mengucapkan teks.

3.4 Metode Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak sistem ini akan menggunakan metode *Waterfall*. Untuk tahap pengembangan, model *waterfall* merupakan model yang paling sering digunakan. Bentuk air terjun juga dikenal dengan model klasik atau konvensional. Model air terjun kadang-kadang dikenal sebagai siklus klasik atau model sekuensial linier. Dimulai dengan analisis, desain, pengkodean, pengujian, dan dukungan, model air terjun ini menyediakan strategi alur kehidupan perangkat lunak yang berurutan (Susilo, 2018).



Gambar 3. 9 Ilustrasi Model Waterfall

a. Analisis Kebutuhan

Tahap ini merupakan tahap awal yang digunakan untuk menganalisa kebutuhan apa saja yang dibutuhkan sebagai syarat dalam mengimplementasikan sistem pengolahan citra untuk membaca teks berbasis android menggunakan metode *template matching correlation*.

b. Desain Sistem

Pada penelitian ini proses konversi *Text-to-Speech* dibutuhkan sebuah masukkan, yaitu gambar dari tangkapan kamera *smartphone*. Kemudian gambar tersebut di *crop* per huruf. Langkah selanjutnya melakukan *preprocessing*, segmentasi, normalisasi, ekstraksi ciri, *recognition*.

c. Implementasi

Pada proses implementasi dilakukan proses *Template Matching* untuk mengidentifikasi huruf yang teridentifikasi. Pada tahap ini dilakukan proses mencocokkan huruf yang ditemukan dengan *template* huruf yang ada.

d. Pengujian Sistem

Pada tahap ini dilakukan pengujian sistem secara menyeluruh yang telah dibuat menggunakan bahasa pemrograman Java untuk dilakukan pengimplementasian sistem pengolahan citra digital untuk membaca teks menggunakan metode *Template Matching Correlation*.

e. Penerapan

Pada tahap terakhir ini, setelah tahap pengujian, maka akan dilakukan langkah berikutnya yaitu penerapan dimana program atau sistem digunakan secara keseluruhan.