# BAB III
# METODE PENELITIAN

## 3.1 Identifikasi Masalah

Seiring perkembangan teknologi informasi, pertukaran informasi sangat mudah untuk dilakukan. Berbagai macam informasi dikirimkan setiap harinya tanpa henti. Salah satu jenis informasi yang sering digunakan adalah gambar. Keamanan dari informasi-informasi yang dikirimkan tersebut seringkali tidak diperhatikan, sehingga pencurian informasi menjadi ancaman yang sangat mungkin terjadi. Salah satu cara untuk mencegah hal tersebut adalah dengan menggunakan kriptografi.

Kriptografi dapat mengamankan informasi yang akan dikirim dengan menyandikan isi informasi tersebut. Gambar yang akan dikirimkan dapat diamanakan dengan cara disandikan oleh algoritma kriptografi. Algoritma kriptografi *Affine Cipher* dengan kunci yang dibangkitkan oleh *CSPRNG* berbasis *Chaos* dapat menyandikan nilai-nilai piksel pada gambar dengan baik hanya dengan sebuah kunci bilangan riil dalam rentang 0 sampai 1. Gambar yang telah disandikan dapat dikembalikan lagi seperti semula untuk mengembalikan isi informasi dari gambar.

## 3.2 Model Dasar

Model dasar pada penelitian ini adalah algoritma *Affine Cipher* dan *CSPRNG* berbasis *Chaos*.

### 3.2.1 *Affine Cipher*

*Affine Cipher* umumnya digunakan pada penyandian pesan. Setiap huruf alfabet pada pesan teks yang akan disandikan diubah ke dalam bilangan lalu dienkripsi menggunakan fungsi matematika, lalu dikembalikan lagi dalam bentuk teks. Ruang plainteks dan cipherteks pada penyandian pesan berada pada $\mathbb{Z}_{26}$ sesuai dengan banyaknya alfabet, sehingga nilai $n$ pada kriptosistem *Affine Cipher* adalah 26. Sedangkan pada penyandian gambar, ruang plain gambar dan cipher gambar berada pada $\mathbb{Z}_{256}$ sesuai dengan rentang nilai piksel pada gambar, sehingga nilai $n$

pada kriptosistem *Affine Cipher* adalah 256. Berikut adalah kriptosistem *Affine Cipher* pada kriptografi gambar:

$$\mathcal{P} = \mathcal{C} = \mathbb{Z}_{256}$$

dan

$$\mathcal{K} = \{(m, b) \in \mathbb{Z}_{256} \times \mathbb{Z}_{256} : FPB(m, 256) = 1\}$$

untuk $K = (m, b) \in \mathcal{K}$ , didefinisikan

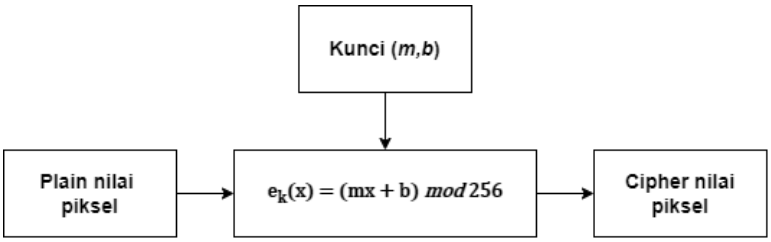
$$e_k(x) = (mx + b)\ mod\ 256 \tag{3.1}$$

dan

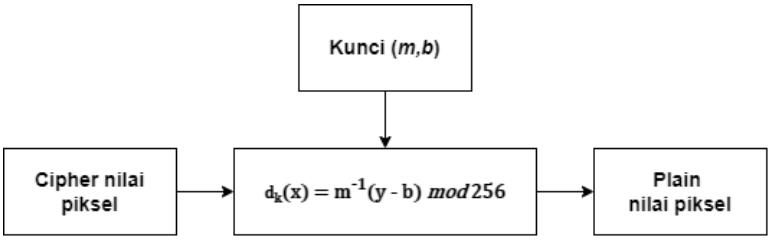
$$d_k(x) = m^{-1}(y - b)\ mod\ 256 \tag{3.2}$$

dengan $x, y \in \mathbb{Z}_{256}$.

*Affine Cipher* hanya membutuhkan sepasang kunci $(m, b)$ untuk melakukan enkripsi atau dekripsi. Gambar 3.1 dan Gambar 3.2 berikut adalah skema enkripsi dan dekripsi *Affine Cipher* pada nilai piksel gambar:




Gambar 3.1 Skema Enkripsi *Affine Cipher* pada Nilai Piksel Gambar




Gambar 3.2 Skema Dekripsi *Affine Cipher* pada Nilai Piksel Gambar

### 3.2.2 *CSPRNG* Berbasis *Chaos*

Fungsi *chaos* yang akan digunakan dalam penelitian ini adalah persamaan logistik pada Persamaan (2.1). *CSPRNG* berbasis *Chaos* membangkitkan barisan bilangan acak berupa bilangan riil dalam rentang 0 sampai 1 dengan masukan nilai awal berupa bilangan riil dalam rentang 0 sampai 1 pula. Barisan bilangan acak yang dihasilkan akan digunakan dalam kriptografi gambar berupa bilangan bulat dalam rentang mulai dari 0 sampai dengan 255. Sehingga barisan bilangan acak

yang dihasilkan *CSPRNG* berbasis *Chaos* perlu dikonversi menjadi barisan bilangan bulat acak dalam rentang mulai dari 0 sampai dengan 255.

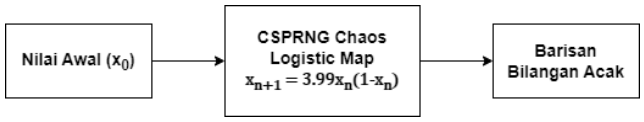
Dalam penelitian ini barisan bilangan riil yang dihasilkan dikonversi dengan menggunakan Persamaan (2.2) dengan $n = 8$. Kemudian barisan bilangan bulat yang didapat dilakukan operasi modulo 256 untuk mendapat barisan bilangan bulat acak dalam rentang mulai dari 0 sampai dengan 255. Sebagai contoh, berdasarkan contoh barisan bilangan acak yang telah dihasilkan pada Subbab 2.2.7, konversi nilai $x_2 = 0{,}981958$ akan menjadi $98195800 \bmod 256$. Begitu pula dengan $x_3 = 0{,}070865$ akan menjadi $7086500 \bmod 256$.

Pada contoh sebelumnya digunakan $r = 4$ pada persamaan logistik untuk membangkitkan barisan bilangan acak. Namun, pemilihan nilai $r = 4$ memiliki dua nilai masukan awal yang menyebabkan barisan bilangan acak yang dihasilkan konvergen ke suatu nilai. Dua nilai yang dimaksud adalah $x_0 = 0{,}5$ dan $x_0 = 0{,}75$. Masing-masing nilai tersebut menyebabkan barisan bilangan acak konvergen ke $0$ dan $0{,}75$. Nilai awal ini akan digunakan sebagai kunci yang akan dipilih oleh pengguna dalam mengenkripsi dan mendekripsi gambar. Kedua nilai awal tersebut sangat memungkinkan untuk dipilih oleh calon pengguna, sehingga diperlukan upaya untuk mengatasi hal ini.

Upaya yang dilakukan dalam penelitian ini adalah dengan mengganti nilai $r = 4$ menjadi $r = 3{,}99$. Meskipun saat nilai $r = 3{,}99$ tidak bersifat *chaos* penuh seperti saat nilai $r = 4$, tetapi sudah sangat mendekati sifat *chaos* penuh. Ketika nilai $r = 3{,}99$ persamaan logistik tetap memiliki satu nilai awal yang menyebabkan barisan bilangan acak yang dihasilkan konvergen ke suatu nilai, yaitu $x_0 = \frac{2{,}99}{3{,}99}$ yang menyebabkan barisan bilangan acak konvergen ke $\frac{2{,}99}{3{,}99}$. Namun, pemilihan kunci $\frac{2{,}99}{3{,}99}$ atau sekitar $0{,}7493734335839599$ akan sangat kecil kemungkinan dipilih oleh calon pengguna. Sehingga kecil juga kemungkinan bagi calon pengguna untuk memilih kunci yang akan menyebabkan barisan bilangan acak yang dihasilkan konvergen ke suatu nilai. Oleh karena itu, dalam penelitian ini digunakan nilai $r = 3{,}99$ dalam parameter persamaan logistik, sehingga persamaan tersebut menjadi

$$x_{n+1} = 3{,}99x_n(1 - x_n) \tag{3.3}$$

Gambar 3.3 berikut adalah skema pembangkitan barisan bilangan acak *CSPRNG* berbasis *Chaos Logistic Map*:




Gambar 3.3 Skema Pembangkitan Barisan Bilangan Acak dengan *CSPRNG* Berbasis *Chaos Logistic Map*

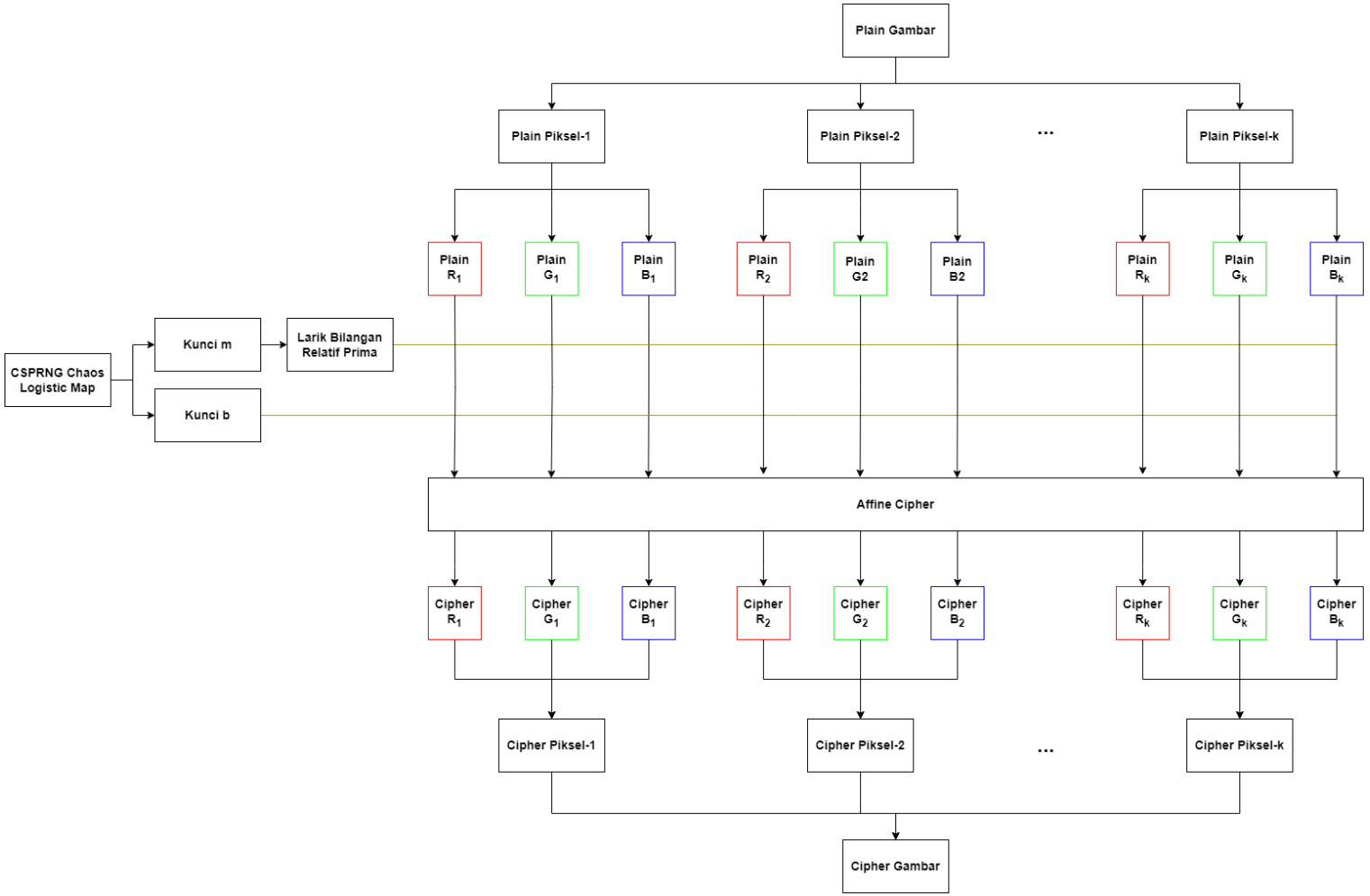
### 3.3 Pengembangan Model Dasar

Pengembangan model dasar pada penelitian ini adalah menggunakan pembangkit bilangan acak *CSPRNG* berbasis *Chaos* untuk membangkitkan barisan bilangan bulat acak untuk digunakan sebagai kunci $m$ dan kunci $b$ yang digunakan dalam *Affine Cipher* untuk mengenkripsi nilai-nilai piksel pada gambar. Sehingga *Affine Cipher* tidak menggunakan hanya sepasang kunci $m$ dan $b$ untuk mengenkripsi seluruh nilai piksel pada gambar, tetapi menggunakan kunci $m$ dan $b$ yang berbeda-beda untuk setiap nilai piksel pada gambar hanya dengan sebuah masukan kunci pada *CSPRNG* berbasis *Chaos*. Barisan kunci $m$ dan kunci $b$ yang dibangkitkan yaitu sebanyak nilai-nilai piksel pada gambar. Gambar *RGB* memiliki ukuran $M \times N \times 3$ sehingga barisan kunci $m$ dan kunci $b$ yang dibangkitkan sebanyak $M * N * 3$. Begitu pula untuk gambar *grayscale* yang memiliki ukuran $M \times N$, barisan kunci $m$ dan kunci $b$ akan dibangkitkan sebanyak $M * N$.

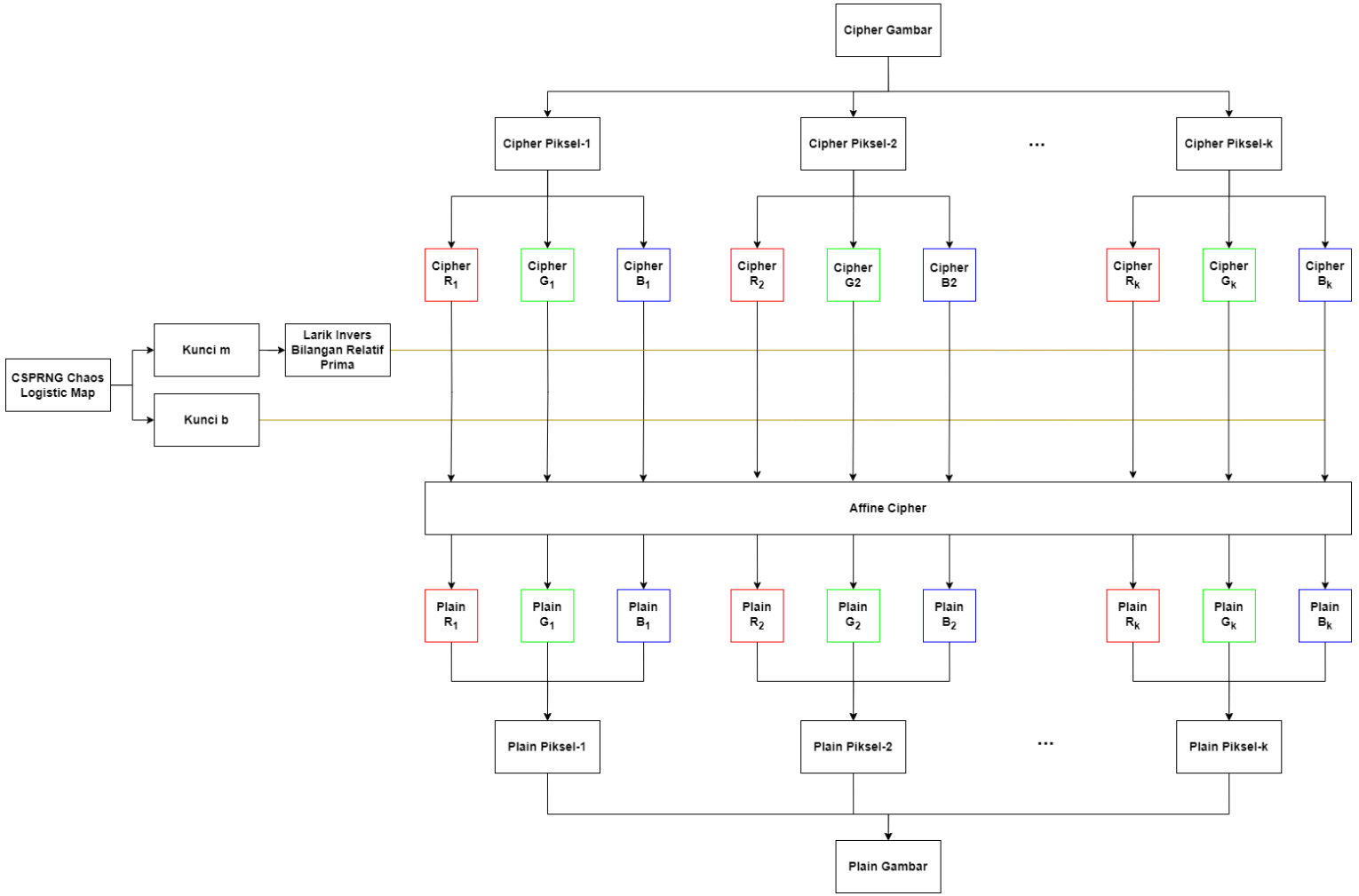
Barisan kunci $m$ dan kunci $b$ yang dibangkitkan berupa bilangan bulat dalam rentang mulai dari 0 sampai dengan 255. Dalam aplikasinya, kunci $m$ haruslah relatif prima dengan $n$, dengan nilai $m$ berada pada rentang mulai dari 0 sampai dengan 255 dan nilai $n$ adalah 256. Sehingga banyaknya nilai $m$ yang relatif prima dengan $n$ adalah 128, begitu pula dengan banyaknya nilai $m^{-1}$ atau invers dari $m$. Dalam pengembangan model ini, nilai-nilai $m$ yang relatif prima dengan $n$ beserta inversnya masing-masing akan disimpan dalam suatu larik. Larik tersebut akan memiliki indeks mulai dari 0 sampai dengan 127. Sehingga untuk pembangkitan kunci $m$ akan dilakukan dengan membangkitkan barisan bilangan bulat acak dalam rentang mulai dari 0 sampai dengan 127 untuk mengakses kunci $m$ dalam larik tersebut. Hal ini dapat dilakukan dengan melakukan operasi modulo

128 pada barisan bilangan bulat yang telah didapat. Sedangkan untuk kunci $b$, tetap menggunakan operasi modulo 256.

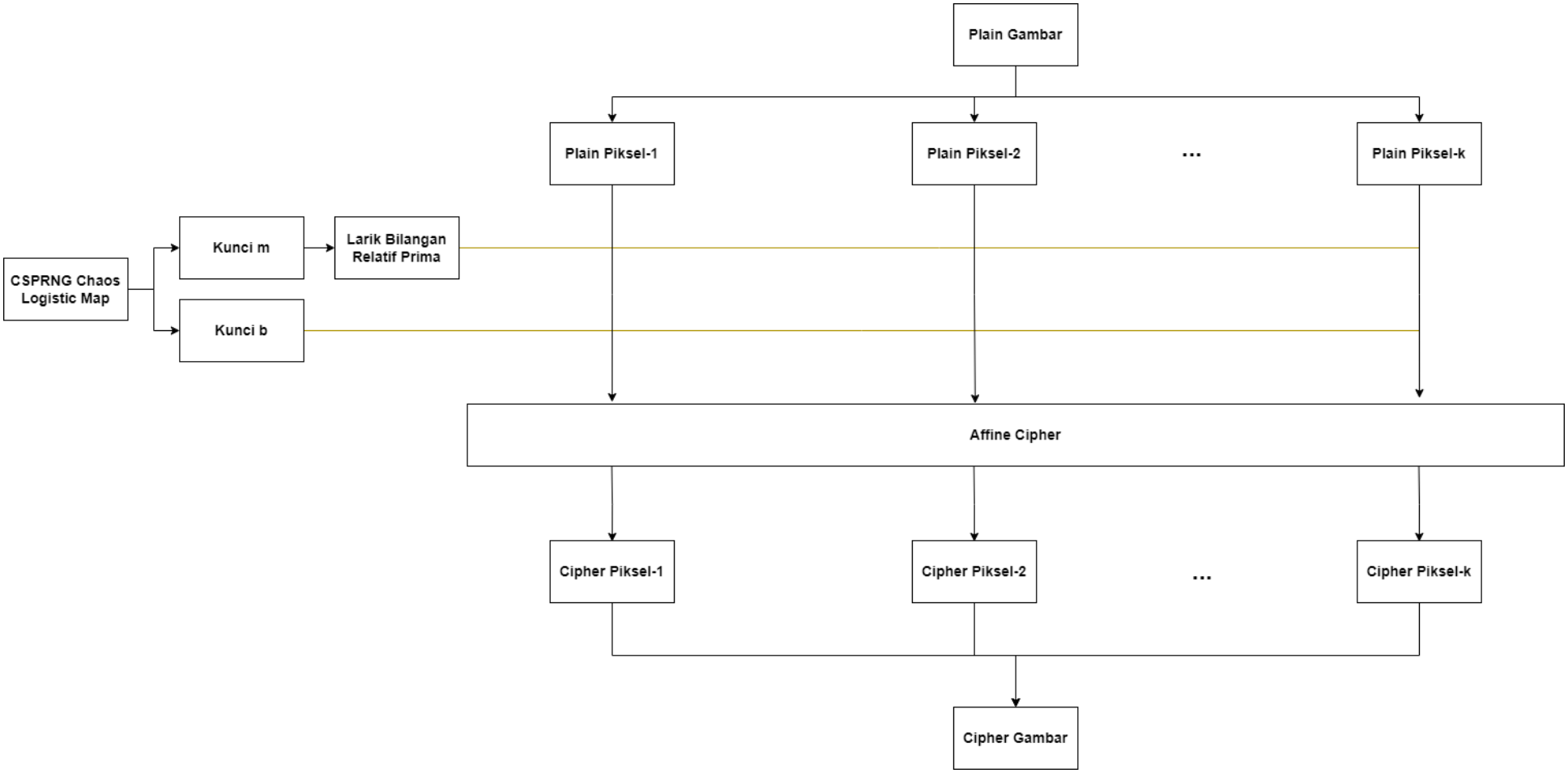
Skema enkripsi dan dekripsi pengembangan model dasar pada gambar RGB disajikan pada Gambar 3.4 dan Gambar 3.5, sedangkan pada gambar *grayscale* disajikan pada Gambar 3.6 dan Gambar 4.7 berikut, dengan $k$ adalah banyaknya piksel pada gambar:



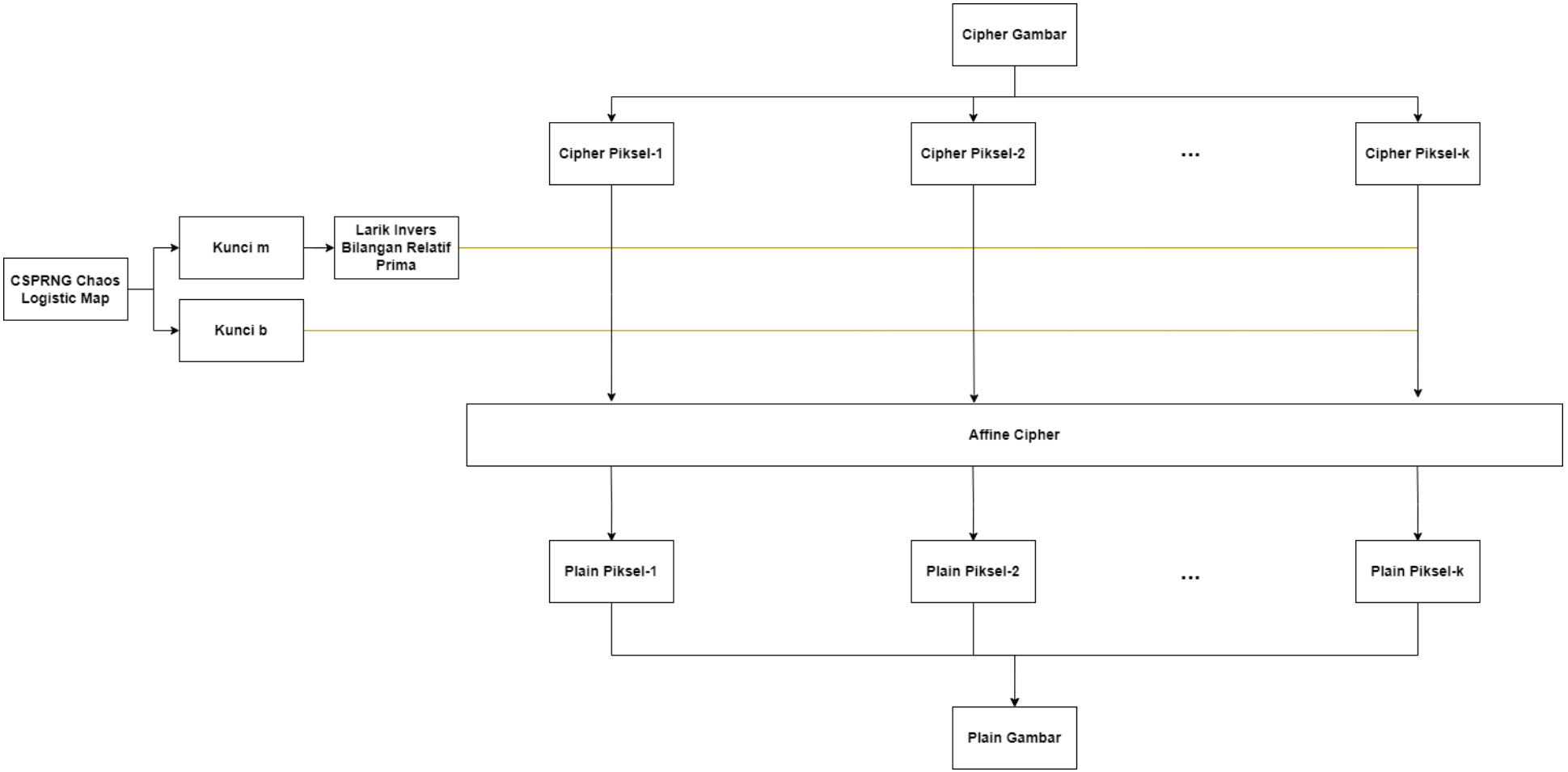

Gambar 3.4 Skema Enkripsi Pengembangan Model Dasar *Affine Cipher* dan *CSPRNG* Berbasis *Chaos* pada Gambar RGB




Gambar 3.5 Skema Dekripsi Pengembangan Model Dasar *Affine Cipher* dan *CSPRNG* Berbasis *Chaos* pada Gambar RGB




Gambar 3.6 Skema Enkripsi Pengembangan Model Dasar *Affine Cipher* dan *CSPRNG* Berbasis *Chaos* pada Gambar *Grayscale*




Gambar 3.7 Skema Dekripsi Pengembangan Model Dasar *Affine Cipher* dan *CSPRNG* Berbasis *Chaos* pada Gambar *Grayscale*

### 3.4 Konstruksi Program Aplikasi

#### 3.4.1 *Input* dan *Output*

Dalam program aplikasi yang akan dibuat, *input* dari enkripsi dan dekripsi berupa berkas gambar yang akan dienkripsi atau didekripsi dalam format JPEG, JPG, atau PNG. Selain itu, *input* kunci berupa bilangan riil antara 0 sampai 1 dibutuhkan baik untuk enkripsi maupun dekripsi. Panjang maksimal kunci adalah 16-digit karena bilangan *floating-point* dalam *Python* memiliki presisi sekitar $15-16$ digit desimal. Sedangkan *output* yang dihasilkan baik dari enkripsi maupun dekripsi berupa berkas gambar hasil enkripsi atau dekripsi dalam format PNG.

#### 3.4.2 Algoritma Deskriptif

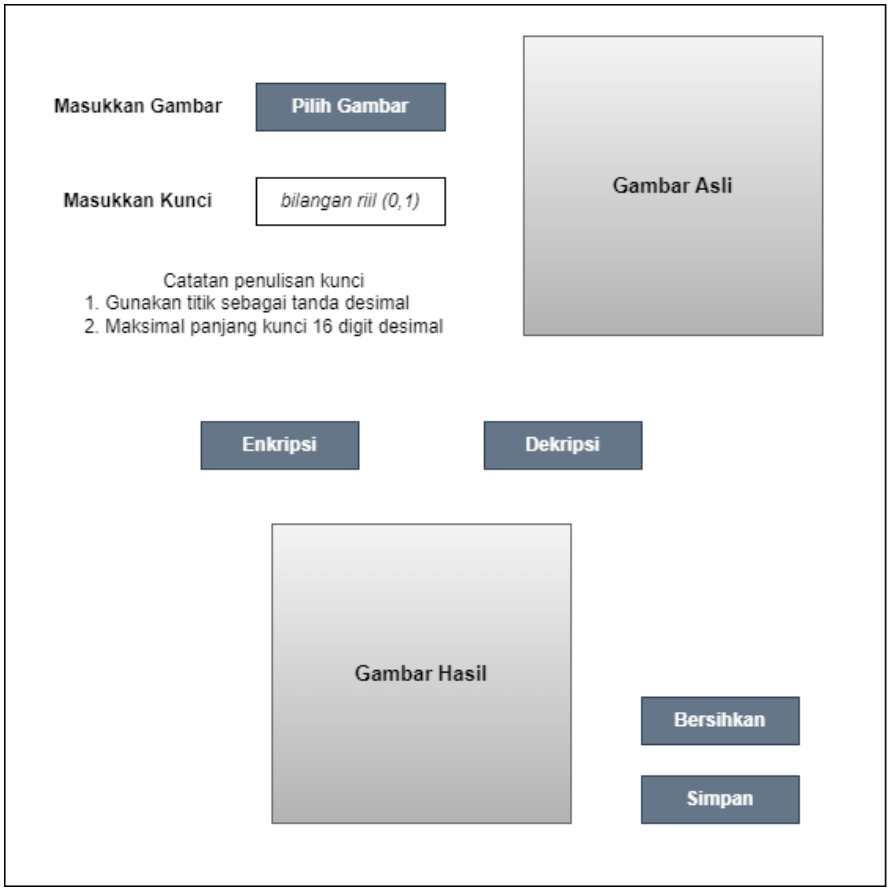
Berikut algoritma Enkripsi dan Dekripsi menggunakan *Affine Cipher* dan *CSPRNG* berbasis *Chaos*:

1. **Algoritma Enkripsi**
   a) Baca gambar sebagai larik dengan menggunakan *library PIL* dan *numpy.*
   b) Ubah larik menjadi 1 dimensi untuk memudahkan komputasi.
   c) Buat larik bilangan-bilangan relatif prima dengan $n$, $n$ bernilai 256, yang akan digunakan sebagai kunci $m$.
   d) Bangkitkan sebuah barisan bilangan acak dengan sebuah kunci menggunakan *CSPRNG* berbasis *Chaos* sepanjang larik gambar dan simpan dalam sebuah larik. Lalu konversi barisan bilangan acak tersebut untuk menghasilkan dua buah barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 127 dan 0 sampai dengan 255.
   e) Gunakan barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 127 untuk mengakses nilai dalam larik bilangan relatif prima yang telah dibuat sebelumnya sebagai kunci $m$ *Affine Cipher*. Dan gunakan barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 255 sebagai kunci $b$ *Affine Cipher.*
   f) Enkripsi nilai-nilai piksel dalam larik gambar dengan menggunakan *Affine Cipher*.
   g) Ubah kembali larik gambar ke dalam dimensi semula.
2. **Algoritma Dekripsi**

a) Baca gambar sebagai larik dengan menggunakan *library PIL* dan *numpy.*
b) Ubah larik menjadi 1 dimensi untuk memudahkan komputasi.
c) Buat larik invers modulo dari bilangan-bilangan yang relatif prima dengan $n$, $n$ bernilai 256, yang akan digunakan sebagai kunci $m$ dalam dekripsi.
d) Bangkitkan sebuah barisan bilangan acak dengan kunci yang sama saat melakukan enkripsi dengan menggunakan *CSPRNG* berbasis *Chaos* sepanjang larik gambar dan simpan dalam sebuah larik. Lalu konversi barisan bilangan acak tersebut untuk menghasilkan dua buah barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 127 dan 0 sampai dengan 255
e) Gunakan barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 127 untuk mengakses nilai dalam larik invers modulo dari bilangan relatif prima yang telah dibuat sebelumnya sebagai kunci $m$ *Affine Cipher.* Dan gunakan barisan bilangan bulat dalam rentang mulai dari 0 sampai dengan 255 sebagai kunci $b$ *Affine Cipher*.
f) Dekripsi nilai-nilai piksel dalam larik gambar dengan menggunakan *Affine Cipher*.
g) Ubah kembali larik gambar ke dalam dimensi semula.

### 3.4.3 Desain Tampilan

Gambar 3.8 berikut adalah desain tampilan program enkripsi dan dekripsi dalam penelitian ini:




Gambar 3.8 Rancangan Tampilan Program Enkripsi dan Dekripsi

### 3.4.4 *Library* Program

Terdapat beberapa *library* dari *python* yang digunakan dalam pemrograman yang dilakukan. Berikut *library* yang digunakan dalam penelitian ini:

1) PIL

   *Library* ini digunakan untuk membaca gambar.

2) Numpy

   *Library* ini digunakan untuk menyimpan gambar sebagai larik. Selain itu, numpy membantu untuk operasi matematika dalam bentuk larik sehingga dapat mengurangi waktu komputasi.

3) TKinter

   *Library* ini digunakan untuk membuat *graphical user interface* (*GUI*) untuk memudahkan pengguna dalam menggunakan program.

### 3.5 Proses Validasi

Proses validasi dilakukan dengan membandingkan larik gambar sebelum dienkripsi dengan larik gambar hasil dekripsi dari gambar yang telah dienkripsi. Pengujian dilakukan dengan masing-masing dua buah gambar RGB dan *grayscale*.

### 3.6 Penarikan Kesimpukan

Pengambilan kesimpulan akan dilakukan berdasarkan hasil validasi dan analisis kualitas enkripsi gambar. Selain itu, akan dibandingkan juga hasil dari pengembangan model dasar dalam penelitian ini dengan hasil dari model dasar.