BAB IV

HASIL DAN PEMBAHASAN

3.1 Hasil Penelitian

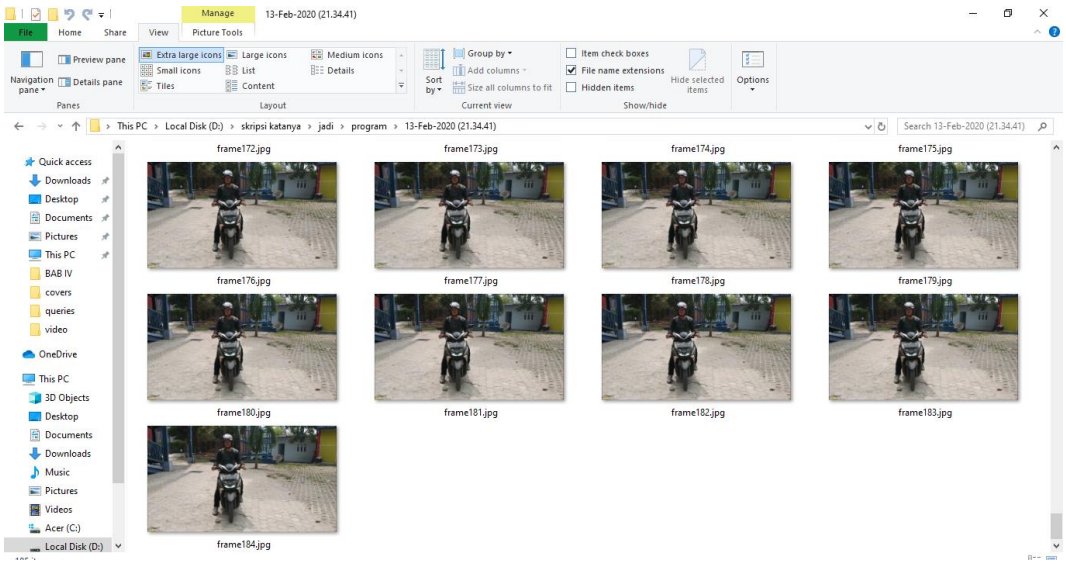
Setelah melalui proses tahapan pengumpulan data dan pembangunan serta metode pengembangan perangkat lunak, maka dihasilkan sebuah aplikasi untuk menverifikasi pengendara serta kendaraannya dengan menggunakan metode *Template Matching* yang dapat digunakan untuk membantu sistem parkir dengan medeteksi seluruh bagian pengendara serta kendaraannya tanpa harus mendeteksi bagian spesifik dari objek citra seperti wajah atau plat kendaraan.

Aplikasi yang dihasilkan merupakan aplikasi konsol dengan ekstensi berkas aplikasi .py , yaitu berkas dapat dijalankan pada IDLE python yang sudah terinstall packages serta library Open CV yang dibutuhkan. Aplikasi ini tidak memiliki tampilan GUI (*Graphical User Interface*) dan saat menjalankan program harus mengetikan nama file secara manual pada IDLE JetBrains PyCharm yang mana saat melakukan eksekusi program dengan menggunakan *command line* yang terdapat pada editor yang digunakan dalam pembuatan aplikasi ini.

4.1.1 Ekstrak Frame

Pada tahap percobaan penelitian ini dilakukan proses ekstrak frame pada sample video masuk. Sebelum proses ekstrak dilakukan, video sample akan di*play*, kemudian proses ekstrak akan berjalan ketika pengendara dan kendaraanya berhenti di depan kamera.

Berikut tampilan hasil proses ekstrak yang dilakukan pada video.



Gambar 4. 1 Ekstrak Frame Video

4.1.2 *Image Preprocessing*

Pada proses *image preprocessing* dilakukan dengan proses *cropping* sehingga dapat mengambil daerah yang diinginkan (ROI) yang akan dijadikan sebagai objek citra. Frame yang akan digunakan adalah frame terakhir dari hasil ekstrak. *Cropping* dilakukan pada saat citra pada pengendara dan kendaraan menghadap ke arah depan kamera yang telah ditandai batasan pada area berupa tempat berhentinya kendaraan saat masuk dan keluar area parkir.

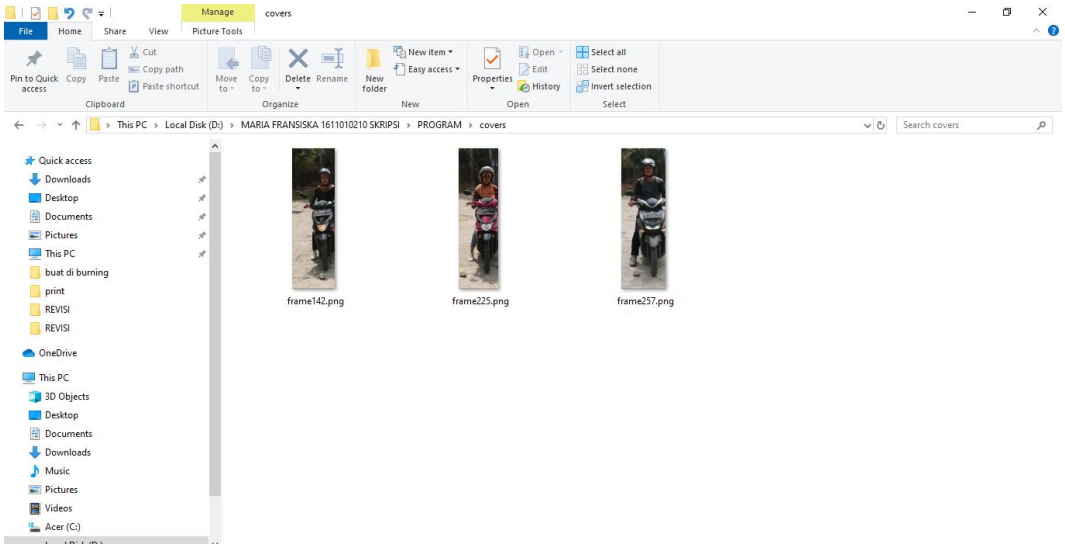
Pada proses *cropping* dimensi atau ukuran citra dari hasil ekstrak frame *capture* video masuk yang digunakan adalah 1920 x 1080 *pixel*, yang kemudian dilakukan cropping pada citra tersebut sehingga ukurannya menjadi 180 x 480 *pixel* untuk citra template. Citra input untuk mencocokannya akan menggunakan dimensi citra 1920 x 1080. Citra input yang digunakan tidak mengalami proses *cropping*. *Cropping* memotong daerah bagian kiri dan kanan frame sehingga didapat bagian daerah tengah, area ini disebut sebagai ROI dalam prosesnya, dan digunakan sebagai template yang kemudian akan

disimpan ke database template. Pada gambar 4.2. merupakan tampilan citra yang diambil dari frame video pengdendara dan kendaraan masuk area parker, yang digunakan sebagai citra template dan gambar 4.3 merupakan citra template yang telah melalui proses *cropping*.

Gambar 4. 2 Frame Template

Gambar 4. 3 *Cropping* Citra Template

Setelah melalui proses *cropping*, citra-citra telah diambil akan disimpan di database template yang digunakan saat matcher mencari kehomogenan dengan citra input. Berikut ini adalah tampilan *database* template dalam penelitian ini.



Gambar 4. 4 Tampilan *Database* Template

4.1.3 Segmentasi

Setelah proses *cropping* dilakukan maka program akan menghasilkan daerah ROI sesuai dengan kebutuhan. **S**egmentasi dilakukan dalam penelitian ini yaitu dengan mendeteksi titik kunci pada gambar yang telah diproses melalui ROI menggunakan algoritma SIFT. Program menggunakan jumlah *ratio* dan minimal *match* yang digunakan dalam menghitung kedekatan jarak *euclidean* pada citra template yang telah disimpan dengan citra input yang didapatkan pada saat kendaaran keluar dari area parkir. Titik kunci yang akan dideteksi yaitu citra template dan citra input.

4.1.4 *Template Matching*

Pada proses ini citra yang sudah didapat sebagai template akan dicocokan atau dicari kesamaannya dengan citra yang didapat pada saat pengendara keluar dari area parkir. Citra tersebut didapat dari sample video kendaraan keluar dari area parkir. Proses pengambilan citra input dilakukan melalui proses yang sama dengan citra template yaitu dengan melakukan ekstrak frame dan mengambil frame terakhir atau frame ketika pengendara dan kendaraan menghadap ke arah depan dengan

posisi tanpa adanya gerakan atau dalam keadaan diam serta tidak dilakukan proses *cropping*.

Dalam proses uji coba ini terdapat tiga komponen opsional yang akan digunakan sebagai faktor yang mempengaruhi hasil dari pengolahan aplikasi seberapa akuratnya yaitu :

1. Rasio
   Rasio jarak tetangga terdekat untuk memangkas jumlah penekanan tombol yang harus dihitung dengan homografi.
2. minMatches
   Jumlah minimum kecocokan yang diperlukan untuk homografi yang akan dihitung.
3. useHamming
   Hamming atau jarak *Euclidean* digunakan untuk membandingkan fitur vektor.

Berikut ini gambar 4.5 merupakan citra input yang akan dicocokan dengan citra template yang sudah tersimpan, dimana pengendara adalah orang sama dan menggunakan kendaraan yang sama.

Gambar 4. 5 Citra Input

Setelah mendapatkan citra input, akan dilanjutkan dengan proses *template matching* yang mencari ada tidaknya kesamaan titik kunci dengan citra template yang telah tersimpan. Jika ditemukan kehomogenan maka program akan menampilkan sebuah *message box* berwarna hijau yang menunjukkan bahwa pengendara tersebut *matching* dengan template yang ada di *database* dan pengendara bisa meninggalkan area parkir. Hal tersebut menyatakan bahwa kendaraan yang dibawa oleh pengendara tersebut adalah miliknya.

Jika tidak ditemukan kehomogenan dari template yang ada maka program akan menampilkan *message box* berwarna merah yang menunjukan bahwa tidak ditemukan kecocokan dengan citra template yang ada dalam database, sehingga pengendara tidak dapat meinggalkan area parkir. Hal tersebut menyatakan bahwa pengendara tersebut bukan pemilik dari kendaraan yang dibawanya.

Hasil percobaan pertama pada program adalah *matching*, karena pengendara dan kendaraan yang digunakan sama saat masuk dan keluar dari area parkir.

Uji coba pertama dilakukan pada pengendara pertama dengan motor yang sama saat masuk dan keluar dari area parkir. Pada percobaan pertama setting yang digunakan adalah dengan menghitung jarak pada nilai *ratio* sebesar 0.9 dan *minMatches* sebesar 270. Hasil yang didapat adalah *matching* karena adanya kehomogenan citra input dengan citra template sehingga menampilkan *message box* berwarna hijau yang menyatakan *match* dan sukses. Berikut hasil percobaan pertama.

Gambar 4. 6 Hasil *Output Matching*

Uji coba kedua dilakukan pada pengendara yang keluar dari area parkir tidak sama dengan kendaraan yang dibawanya saat keluar dari area parkir. Pada percobaan ini setting yang digunakan adalah dengan menghitung jarak pada nilai *ratio* sebesar 0.9 dan *minMatches* sebesar 270. Hasil

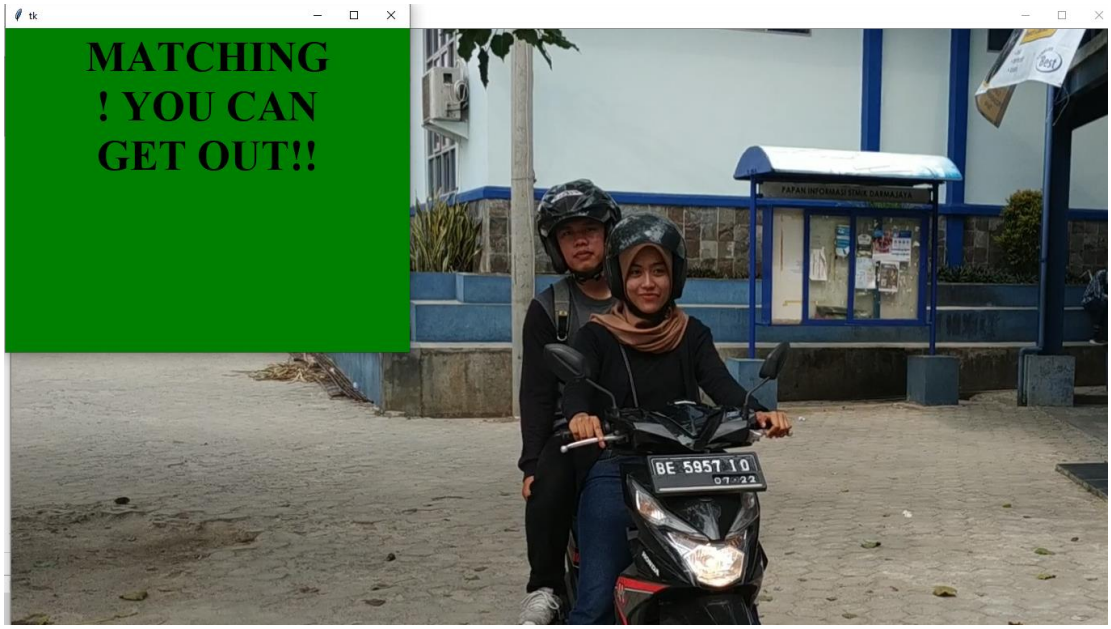
yang didapat adalah tidak *match* karena tidak ditemukan kehomogen antara citra input dan citra template. Hal tersebut juga dikarenakan pengendaranya yang berbeda tetapi motornya sama serta menunjukan bahwa pengendara tersebut bukan pemilik kendaran yang dibawanya. Berikut tampilan citra template dan citra input percobaan kedua dengan motor yang sama tetapi pengendaranya berbeda :

Gambar 4. 7 Citra Template Uji Coba Kedua

Gambar 4. 8 Citra Input Uji Coba Kedua

Gambar 4. 9 Hasil *Output* Tidak *Matching*

Percobaan ketiga dilakukan pada pengendara yang keluar dari area parkir dengan kendaraan yang dibawanya saat keluar dari area parkir, namun berboncengan. Pada percobaan ini setting yang digunakan adalah dengan menghitung jarak pada nilai *ratio* sebesar 0.9 dan *minMatches* sebesar 270. Hasil yang didapat adalah *matching* karena terdapat kehomogenan antara citra input dan citra template. Hal tersebut juga dikarenakan pengendaranya yang masuk dan keluar adalah sama atau pemiliknya yang membedakan hanya jumlah orang pada kendaraan yang digunakan. Pada awal pengendara hanya masuk seorang diri dan saat keluar membonceng rekannya sehingga hasilnya tetap *matching* karena memang sesuai dengan template yang ada. Pengendaraa yang berboncengan tidak mempengaruhi hasil *matcher* dikarenakan titik kunci dari citra input dengan citra template mempunyai banyak kehomogenan maka hasil yang didapat masih tetap *match*. Berikut tampilan citra template dan citra input percobaan ketiga dengan motorvdan pengendara yang berboncengan :



Gambar 4. 10 Hasil *Output* Pengendara Berboncengan

4.2 Pembahasan

Pengujian dilakukan dengan mengubah nilai pada *ratio* dan minMatches pada program. Pengendara dalam sample hanya ada 3 pengendara dan 3 motor. Pengujian tidak hanya dilakukan pada saat pengambilan video dari tampak depan tetapi juga dilakukan dengan mengambil sample video dari tampak samping.

Pengujian pertama dilakukan pada *ratio* = 0.9 dan minMatches=270 pada pengambil sample tampak depan didapatkan hasil sebagai berikut :

Tabel 4. 1 Pengujian Awal Tampak Depan

Pengendara Masuk Area	**Pengendara Keluar**	**Kendaraan**	**Hasil**
Pengedara 1	Pengendara 1	Sama	Matching
Pengendara 2	Pengendara 2	Sama	Matching
Pengendara 3	Pengendara 3	Sama	Matching
Pengendara 1	Pengendara 2	Sama	Not Match
Pengendara 3	Berboncengan	Sama	Matching

Pengujian pertama dilakukan pada *ratio* = 0.9 dan minMatches=270 pada pengambilan sample tampak samping didapatkan hasil sebagai berikut :

Tabel 4. 2 Pengujian Awal Tampak Samping

Pengendara Masuk Area	Pengendara Keluar	Kendaraan	Hasil
Pengedara 1	Pengendara 1	Sama	Matching
Pengendara 2	Pengendara 2	Sama	Matching
Pengendara 3	Pengendara 3	Sama	Matching
Pengendara 2	Pengendara 3	Sama	Matching
Berboncengan	Pengendara 1	Sama	Matching

Pengujian kedua dilakukan dengan mengurangi nilai *ratio* dan minMatches menjadi *ratio* =0.5 dan MinMatches = 200. Berikut ini hasil pengujian kedua dengan pengambilan sample dari tampak depan :

Tabel 4. 3 Pengujian Kedua Tampak Depan

Pengendara Masuk Area	Pengendara Keluar	Kendaraan	Hasil
Pengedara 1	Pengendara 1	Sama	Not Match
Pengendara 2	Pengendara 2	Sama	Not Match
Pengendara 3	Pengendara 3	Sama	Not Match
Pengendara 1	Pengendara 2	Sama	Not Match
Pengendara 3	Berboncengan	Sama	Not Match

Berikut ini hasil pengujian kedua dengan pengambilan sample dari tampak samping :

Tabel 4. 4 Pengujian Kedua Tampak Samping

Pengendara Masuk Area	**Pengendara Keluar**	**Kendaraan**	**Hasil**
Pengedara 1	Pengendara 1	Sama	Not Match
Pengendara 2	Pengendara 2	Sama	Not Match
Pengendara 3	Pengendara 3	Sama	Not Match
Pengendara 2	Pengendara 3	Sama	Not Match
Berboncengan	Pengendara 1	Sama	Not Match

Pengujian ketiga dilakukan dengan menaikkan nilai *ratio* dan minMatches pada program lebih dari nilai *ratio* dan minmaches pada penngujian pertama. Pengujian selanjutnya dilakukan dengan nilai *Ratio* = 1.5 dan minMatches = 350. Berikut ini hasil pengujian ketiga dengan pengambilan sample dari tampak depan :

Tabel 4. 5 Pengujian Ketiga Tampak Depan

Pengendara Masuk Area	**Pengendara Keluar**	**Kendaraan**	**Hasil**
Pengedara 1	Pengendara 1	Sama	Matching
Pengendara 2	Pengendara 2	Sama	Matching
Pengendara 3	Pengendara 3	Sama	Matching
Pengendara 1	Pengendara 2	Sama	Matching
Pengendara 3	Berboncengan	Sama	Matching

Berikut ini hasil pengujian ketiga dengan pengambilan sample dari tampak samping :

Tabel 4. 6 Pengujian Ketiga Tampak Samping

Pengendara Masuk Area	**Pengendara Keluar**	**Kendaraan**	**Hasil**
Pengedara 1	Pengendara 1	Sama	Matching
Pengendara 2	Pengendara 2	Sama	Matching
Pengendara 3	Pengendara 3	Sama	Matching
Pengendara 2	Pengendara 3	Sama	Matching
Berboncengan	Pengendara 1	Sama	Matching

Pengujian pada sample tidak hanya dilakukan dari tampak depan tetapi dalam penelitian ini juga dilakukan dengan mengambil video saat pengendara masuk dan keluar dengan sudut pengambilan video dari arah samping.

Hasil pengujian yang dilakukan dari jumlah pengendara serta motor dan sudut pengambilan gambar pada penelitian ini, bahwa algoritma SIFT untuk mendeteksi titik kunci pada citra untuk mencari jarak tedekat (euqlidean distance) dalam metode Template matching bisa digunakan jika dalam mengatur jumlah *ratio* dan minMatches dengan tepat. Jika pengaturan *ratio* dan minMatches lebih rendah nilainya maka semakin kecil kemungkinan untuk mencari kesamaan titik kunci sehingga menghasilkan Not Match. Jika pengaturan *ratio* dan minMatches semakin besar nilainya maka semakin besar kemungkinan mencari kesamaan titik kunci namun hasilnya tidak akurat sehingga menghasilkan matching pada penelitian ini walau pengendara berbeda saat masuk dan keluar serta saat pengendara berboncengan.

Pada saat pengambilan video juga mempengaruhi hasil matching, sehingga pengambilan sample video dari samping tidak begitu akurat dari hasil pengujian

di atas dibandingkan pengambilan sample video dari depan menghasilkan hasil yang akurat.

Pengujian dilakukan dengan mengambil template dengan cara *cropping* tidak secara sempurna yaitu memotong citra template dengan tidak mengambil bagian secara keseluruhan tetapi menghilangkan sedikit bagian dari target objek, seperti gambar berikut ini *cropping* dilakukan dengan menampilkan pengendara lengkap dengan kendaraannya namun bagian spion motor tidak terambil semua :

Gambar 4. 11 Pengujian *Crop* Template

Hasil yang didapat saat pengujian *crop* template tidak mengambil ROI yang sesuai :

Tabel 4. 7 Pengujian *Crop* Template

Pengendara Masuk Area	**Pengendara Keluar**	**Kendaraan**	**Hasil**
Pengedara 1	Pengendara 1	Sama	Not Match
Pengendara 2	Pengendara 2	Sama	Not Match
Pengendara 3	Pengendara 3	Sama	Matching

Kelebihan dari program aplikasi ini adalah sebagai berikut :

1. Melakukan deteksi secara keseluruhan bagian pada target yaitu pengendara serta kendaraannya tanpa harus mendeteksi secara spesifik pada bagian wajah atau plat kendaraan.
2. Aplikasi ini juga mampu mendeteksi titik kunci lain pada citra yang lebih sederhana yaitu pada objek bukan pada manusia atau kendaraan.

Kekurangan dari program aplikasi ini adalah sebagai berikut :

1. Aplikasi ini sangat mempengaruhi hasil matcing pada saat menentukan nilai *ratio* dan minMatches pada program.
2. Penggunaan aplikasi ini saat objek diambil tampak samping mengalami beberapa kesalahan saat mendeteksi.
3. Aplikasi ini harus mengambil objek dari tampak depan serta pengambilan template dengan melakukan *crop* secara benar mengambil seluruh bagian penting objek tanpa ada perpotongan yang mengurangi bentuk objek tersebut.
4. Aplikasi ini dalam menjalankan eksekusi *source code* harus melalui *command line* dan masih berbentuk konsol.