BAB IV
ANALISIS DAN PEMBAHASAN

4.1 Pengujian Fungsionalitas Fitur-Fitur pada Sistem

4.1.1 Pengujian Fitur *Tuning* Warna HSV

4.1.1.1 *Tuning* Warna HSV Lapangan

Fitur *tuning* warna lapangan hanya dapat diakses dengan menekan huruf "f" (kecil), apabila diketikkan dengan "F" (kapital), maka fitur ini tidak akan muncul. Setelah perintah diberikan, maka akan muncul tampilan *masking* atau *thresholding* warna hijau lapangan dan *scroll bar* untuk mengatur nilai parameternya.

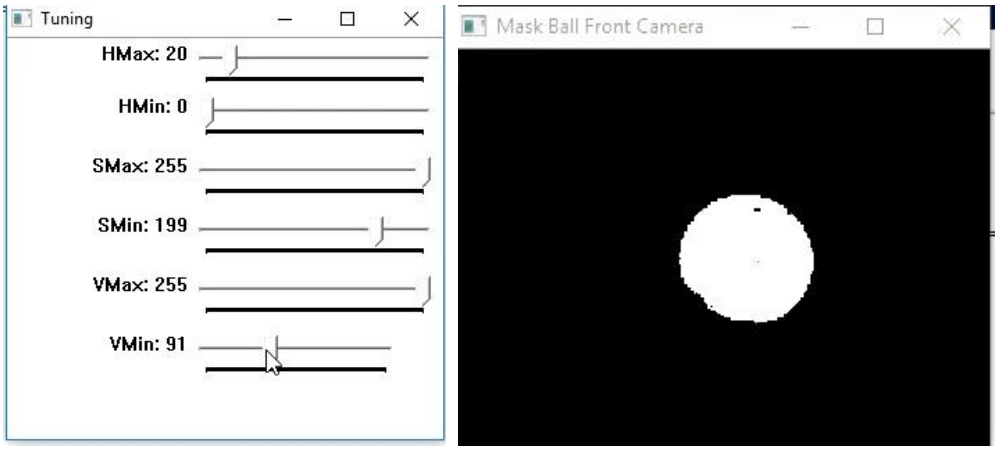
Gambar 4.1 *Tuning* Warna Lapangan

Terlihat pada **Gambar 4.1**, bahwa fitur *tuning* warna lapangan telah dapat berjalan dengan baik, dimana dalam pengujian tersebut berhasil didapatkan nilai parameter batas atas dan bawah warna HSV lapangan. Selain itu, saat *scroll bar* berubah posisinya, nilai parameternya pun ikut berubah, serta diikuti dengan perubahan tampilan pada *masking* warna hijau lapangan. Indikator-indikator tersebut menunjukkan bahwa fitur *tuning* warna lapangan telah dapat berjalan dengan baik.

4.1.1.2 *Tuning* Warna HSV Bola pada Kamera Depan

Fitur *tuning* nilai parameter warna HSV bola pada kamera depan dapat dijalankan dengan menekan tombol "b" pada *keyboard*. Perintah ini juga tidak

dapat diaktifkan dengan menggunakan huruf "B" (kapital). Seperti halnya pada fitur *tuning* warna HSV lapangan, setelah diberikan perintah, maka akan muncul tampilan *masking* atau *thresholding* warna bola dan *scroll bar* untuk mengatur nilai parameternya.

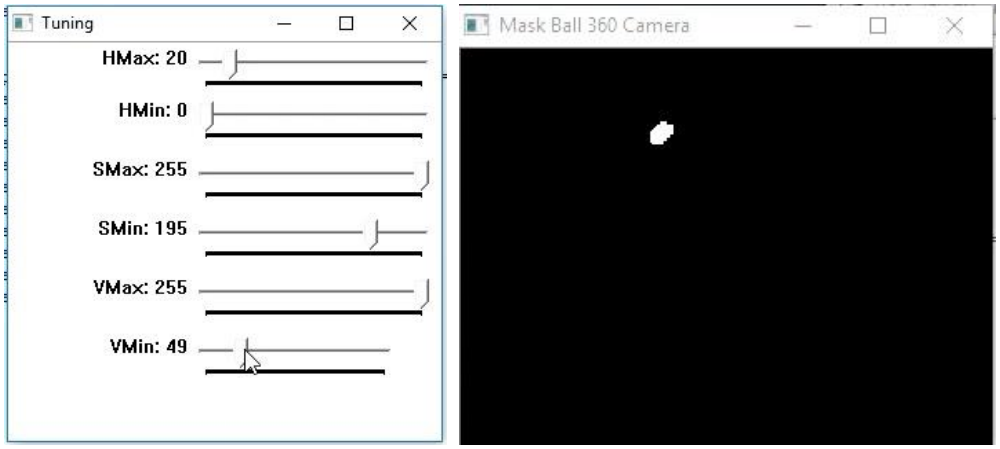


Gambar 4.2 *Tuning* Warna Bola pada Kamera Depan

Berdasarkan **Gambar 4.2**, terlihat bahwa tampilan *masking* warna bola dan nilai parameter HSV dapat berubah sesuai dengan perubahan posisi *scroll bar.* Selain itu, pada pengujian tersebut juga berhasil didapatkan nilai parameter batas atas dan bawah warna HSV bola pada kamera depan. Indikator-indikator tersebut menunjukkan bahwa fitur *tuning* warna bola pada kamera depan telah dapat berjalan dengan baik, sehingga apabila ingin merubah nilai parameter HSVnya tidak perlu melalui *coding*, tetapi cukup dengan menggeser-geser *scroll bar.*

4.1.1.3 *Tuning* Warna HSV Bola pada Kamera Omnivision

Fitur *tuning* nilai parameter batas atas dan bawah warna HSV bola pada kamera omnivision tidak menjadi satu dengan kamera depan, hal itu dikarenakan kamera omnivision tidak langsung menghadap pada bola, melainkan mendeteksi bola melalui pantulan bola kaca, dan selain itu juga ter*cover* oleh tabung akrilik pada sisi luar kamera. Hal tersebut mengakibatkan terjadinya sedikit perubahan pada warna bola, sehingga penyetelan parameter HSVnya perlu untuk dibedakan. Fitur *tuning* warna bola pada kamera omnivision dapat dipanggil dengan mengetikkan huruf "t" pada *keyboard*. Seperti halnya fitur lainnya, fitur ini juga tidak dapat dipanggil dengan huruf kapital.



Gambar 4.3 *Tuning* Warna Bola pada Kamera Omnivision

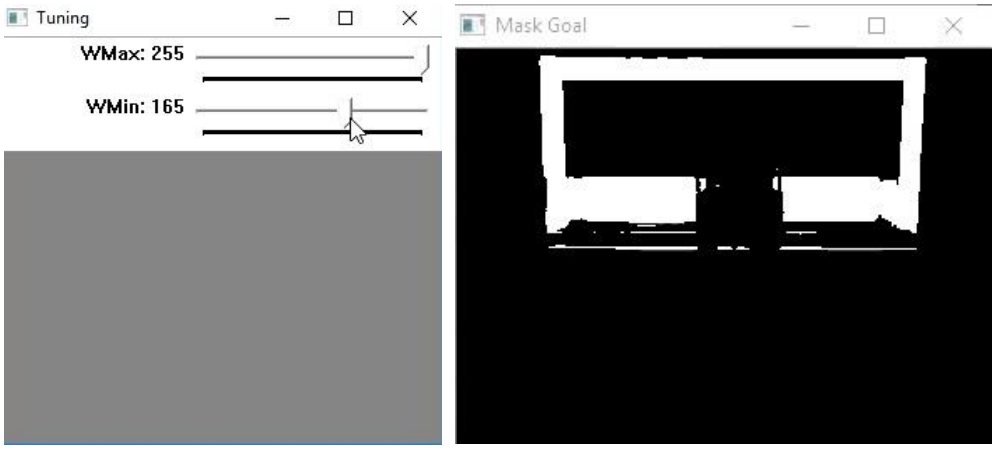
Berdasarkan pada **Gambar 4.3**, terlihat bahwa fitur *tuning* warna bola pada kamera omnivision juga dapat berjalan dengan baik, hal tersebut terlihat dari berhasil didapatkannya nilai parameter batas atas dan bawah warna HSV bola pada kamera omnivision. Selain itu, pada saat *scroll bar* berubah posisinya, nilai parameternya pun ikut berubah, serta diikuti dengan perubahan tampilan pada *masking* warna HSV bola pada kamera omnivision. Indikator-indikator tersebut menunjukkan bahwas fitur *tuning* warna HSV bola pada kamera omnivision telah dapat berjalan dengan baik.

4.1.1.4 *Tuning* Warna *Grayscale* Gawang

Fitur *tuning* warna gawang sedikit berbeda dengan fitur *tuning* objek lainnya, hal ini dikarenakan pada proses deteksi gawang lawan tidak menggunakan parameter HSV, melainkan memakai metode citra *Grayscale.* Jadi, ketika fitur *tuning* warna gawang dipanggil dengan mengetikkan huruf "g" pada *keyboard*, yang muncul pada tampilan *scroll bar* hanya terdapat dua buah parameter saja, tidak seperti HSV yang menggunakan total hingga enam buah parameter. Kedua parameter tersebut akan mengatur nilai intensitas dari citra *Grayscale*, yaitu dimana terbentang pada nilai *range* 0 (hitam) hingga 255(putih).

Berdasarkan hasil pengujian, fitur *tuning* warna *Grayscale* gawang lawan telah dapat berjalan sesuai dengan harapan. Indikator yang menunjukkan keberhasilan tersebut yaitu tampilan *masking* atau *thresholding* warna *Grayscale* gawang lawan dan nilai parameternya telah dapat berubah sesuai dengan pergeseran

posisi *scroll bar*. Selain itu, pada pengujian ini juga berhasil didapatkan nilai parameter batas atas dan bawah warna *Grayscale* gawang lawan, dimana tampilan *masking*nya akan terlihat seperti pada **Gambar 4.4**.

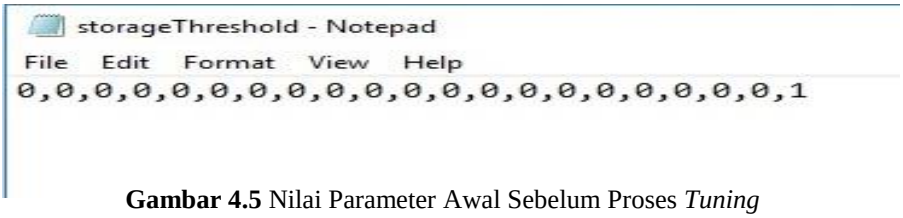


Gambar 4.4 *Tuning* Warna Gawang

4.1.2 Pengujian Fitur *Save* dan *Load*

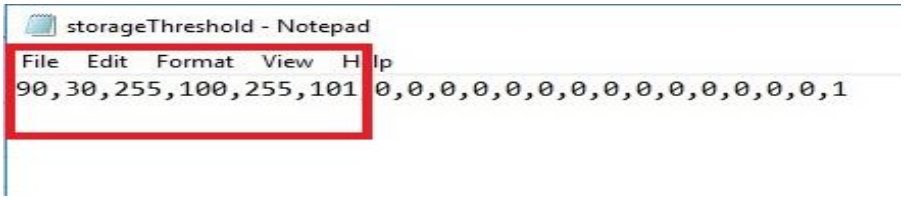
4.1.2.1 Fitur *Save*

Perintah *save* yang telah disematkan di dalam sistem bertujuan untuk menyimpan data-data parameter hasil dari proses *tuning* warna objek yang akan dideteksi. Perintah *save* dapat dijalankan dengan mengetikkan huruf "s" pada *keyboard.* Data-data parameter tersebut kemudian akan tersimpan di dalam sebuah file bernama "*Storage Threshold.txt*".



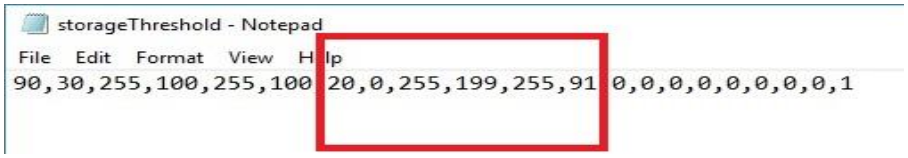
Gambar 4.5 Nilai Parameter Awal Sebelum Proses *Tuning*

Terlihat pada **Gambar 4.5**, nilai parameter sebelum dilakukan proses *tuning* warna objek adalah bernilai nol (0). Jadi dengan demikian, pada saat membuka fitur *tuning* warna objek, semua posisi *track bar* berada pada posisi paling kiri, dimana nilai-nilai parameternya adalah nol (0), serta tampilan *masking* warna akan berwarna hitam secara keseluruhan.



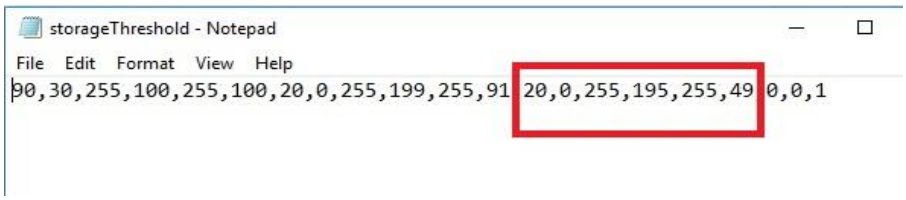
Gambar 4.6 Nilai Parameter Tersimpan Setelah Dilakukan *Tuning* Warna Lapangan

Setelah proses *tuning* warna lapangan selesai dilakukan, kemudian disimpan (*save*), maka nilai parameter yang terdapat di dalam file "*Storage Threshold.txt*" akan berubah menjadi seperti pada **Gambar 4.6**. Hal tersebut menunjukkan bahwasannya data-data nilai parameter batas atas dan bawah warna HSV lapangan telah berhasil tersimpan pada urutan data ke 0 hingga ke 5 di dalam file tersebut.



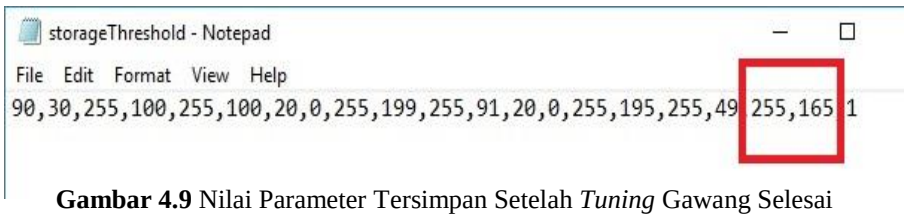
Gambar 4.7 Nilai Parameter Tersimpan Setelah Dilakukan *Tuning* Bola pada Kamera Depan

Pada **Gambar 4.7** terlihat bahwa data ke 6 hingga ke 11 dalam file "*Storage Threshold.txt*" telah terisi oleh data-data nilai parameter batas atas dan bawah warna HSV bola pada kamera depan. Dimana sebelum dilakukan proses *tuning* warna bola pada kamera depan, nilai data ke 6 hingga ke 11 adalah nol. Indikator tersebut menunjukkan bahwasanya perintah *save* pada fitur *tuning* bola pada kamera depan telah dapat berjalan dengan baik.



Gambar 4.8 Nilai Parameter Tersimpan Setelah Dilakukan *Tuning* Bola pada Kamera Omnivision

Pada file "*Storage Threshold.txt*", data-data nilai parameter HSV bola pada kamera omnivision terletak pada urutan ke 12 hingga ke 17. Terlihat dari hasil pengujian pada **Gambar 4.8**, bahwa pada urutan data tersebut telah terisi nilai-nilai parameter HSV bola pada kamera omnivision, hal tersebut menunjukkan bahwa perintah *save* pada fitur *tuning* tersebut telah dapat berjalan dengan baik.



Gambar 4.9 Nilai Parameter Tersimpan Setelah *Tuning* Gawang Selesai

Seperti halnya pada fitur-fitur *tuning* objek lainnya, pada fitur *tuning* gawang pun perintah *save* juga telah berhasil berjalan dengan baik. Hal tersebut terlihat pada **Gambar 4.**9, dimana data ke 18 dan 19 pada file "*Storage Threshold.txt*" telah berganti nilainya setelah dilakukan proses penyetelan nilai parameter batas atas dan bawah warna *Grayscale* gawang lawan.

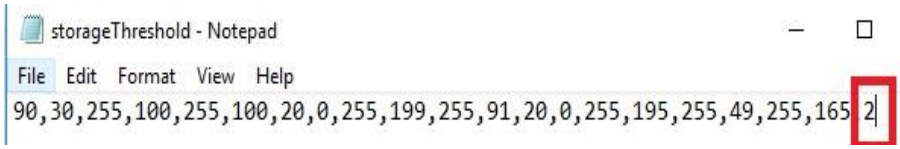
4.1.2.2 Fitur *Load*

Setelah pengujian perintah *save* selesai dilakukan, pengujian selanjutnya adalah pengujian perintah *load.* Berbeda dengan perintah *save*, perintah *load*, memiliki fungsi untuk memanggil data-data parameter yang telah disimpan dalam file "*Storage Threshold.txt*". Perintah *load* akan berjalan secara otomatis ketika sistem *image processing* pada robot sepak bola beroda Mr Dev mulai dijalankan atau dioperasikan.

Berdasarkan hasil pengujian, terlihat bahwa saat fitur-fitur *tuning* dijalankan kembali, posisi masing-masing *scroll bar*, dan tampilan *masking* atau *thresholding* warna telah menunjukkan hasil yang sesuai dengan nilai parameter yang telah tersimpan sebelumnya, hal tersebut akan terlihat seperti pada **gambar 4.1**, **4.2**, **4.3**, dan **4.4**. Indikator tersebut membuktikan bahwasannya perintah *load* telah dapat berjalan dengan baik sesuai dengan harapan.

4.1.3 Pengujian Fitur Pergantian Tanda Warna Lawan

Secara *default* tanda warna lawan pada sistem *image processing* Mr Dev adalah berwarna *cyan*, yang mana diwakilkan dengan nilai satu (1) dan tersimpan di dalam file "*Storage Threshold.txt*" pada urutan data ke 20 atau terakhir. Disisi lain, warna magenta diwakilkan dengan nilai dua (2) dan tersimpan pada urutan data yang sama pada file tersebut.



Gambar 4.10 Nilai Parameter Tersimpan jika Musuh Berwarna Magenta

Berdasarkan **Gambar 4.10**, terlihat bahwa sistem telah berhasil mengganti tanda warna lawan dari yang semula atau *default*nya adalah *cyan*, berubah menjadi magenta. Hal tersebut terbukti dengan berubahnya data pada urutan ke 20 dari yang semula adalah bernilai satu (1) berubah menjadi bernilai dua (2). Setelah program disimpan, kemudian sistem dijalankan kembali, warna nomor punggung lawan tetap magenta atau dengan kata lain berhasil disimpan (*save*) dan dipanggil kembali (*load*) ke dalam sistem.

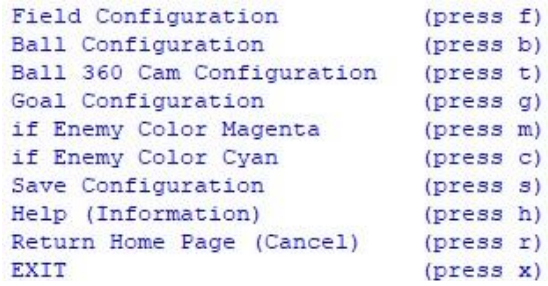
Gambar 4.11 Nilai Parameter Tersimpan jika Musuh Berwarna *Cyan*

Pengujian selanjutnya adalah warna nomor punggung atau tanda warna lawan dikembalikan kembali menjadi warna *cyan*. Pengujian ini menunjukkan hasil yang positif, yaitu dimana tanda warna lawan berhasil berganti dari warna magenta menjadi warna *cyan.* Hal tersebut dapat dibuktikan dengan berubahnya nilai data urutan ke 20 pada file tempat penyimpanan parameter menjadi bernilai satu (1) kembali.

4.1.4 Pengujian Fitur *Help, Return dan Exit*

4.1.4.1 Fitur *Help*

Fitur *help* pada sistem *image processing* robot sepak bola beroda Mr Dev berfungsi untuk memberikan informasi kepada *user* tentang fitur-fitur atau perintah-perintah yang dapat digunakan di dalam sistem. Fitur ini dapat diakses dengan mengetikkan karakter "h" pada *keyboard.*



Gambar 4.12 Tampilan pada Sistem setelah Perintah *Help* Dijalankan

Berdasarkan hasil pengujian, perintah *help* telah dapat berjalan dengan baik, yaitu dimana ketika perintah tersebut dijalankan, maka pada sistem akan muncul informasi daftar fitur-fitur atau perintah yang dapat dijalankan pada sistem. Tampilan informasi tersebut akan terlihat seperti pada **Gambar 4.12**.

4.1.4.2 Fitur *Return*

Fitur ini memiliki fungsi untuk mengembalikan ke tampilan awal program atau *homepage* setelah mengakses *submenu*. Perintah *return* dapat dipanggil dengan mengetikkan karakter "r" pada *keyboard.*

Berdasarkan hasil pengujian, perintah *return* telah dapat berjalan sesuai dengan harapan, yaitu ketika perintah ini dijalankan, maka akan mengembalikan ke tampilan awal program atau *homepage*. Jadi dengan kata lain, ketika perintah ini dijalankan, maka akan menutup semua jendela-jendela tambahan, seperti yang terdapat pada fitur *tuning* warna objek.

4.1.4.3 Fitur *Exit*

Perintah *exit* berfungsi untuk menutup sistem dan juga menonaktifkan kamera *webcam* yang digunakan untuk mengambil citra. Perintah ini dapat dipanggil dengan mengetikkan huruf "x" pada *keyboard*.

Berdasarkan hasil pengujian, perintah *exit* juga telah dapat berjalan dengan baik. Hal tersebut terbukti ketika perintah ini dijalankan, maka sistem akan langsung berhenti bekerja, semua jendela tertutup, dan kedua kamera webcam pun juga ternonaktifkan. Jadi dengan demikian, sistem dapat ternonaktifkan secara sempurna setelah perintah *exit* dijalankan.

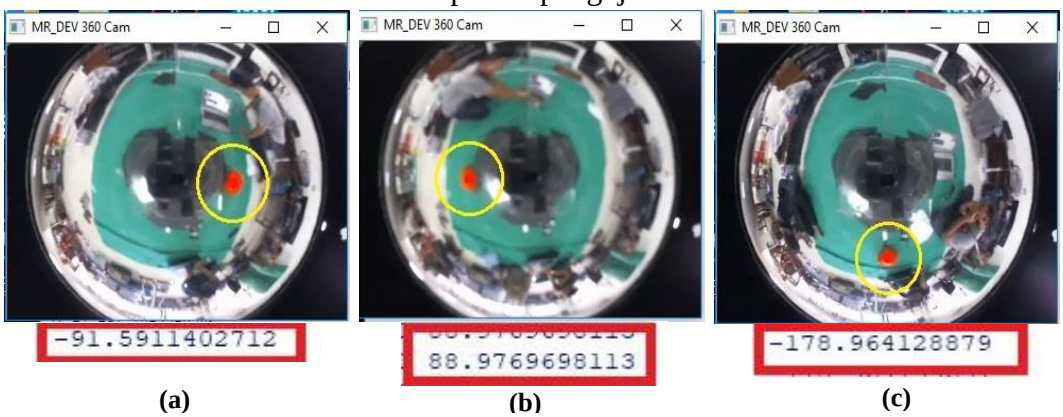
4.2 Pengujian Pembacaan Sudut pada Kamera Omnivision

Berikut adalah data hasil pengujian pembacaan sudut objek terdeteksi terhadap kamera omnivision.

Tabel 4.1 Hasil Pengujian Pembacaan Sudut pada Kamera Omnivision

No	Sudut Objek	Hasil Pembacaan Sudut pada Sistem	Selisih Hasil Pembacaan	*Error*
1	0°	1,48°	1,48°	1,48%
2	45°	44,39°	0,61°	0,61%
3	90°	88,98°	1,02°	1,02%
4	135°	136,01°	1,01°	1,01%
5	180° / -180°	-178,96°	1,04°	1,04%
6	-45°	-43,41°	1,59°	1,59%
7	-90°	-88,98°	1,02°	1,02%
8	-135°	-134,19°	0,81°	0,81%
Rata-Rata			1,0725°	1,0725%

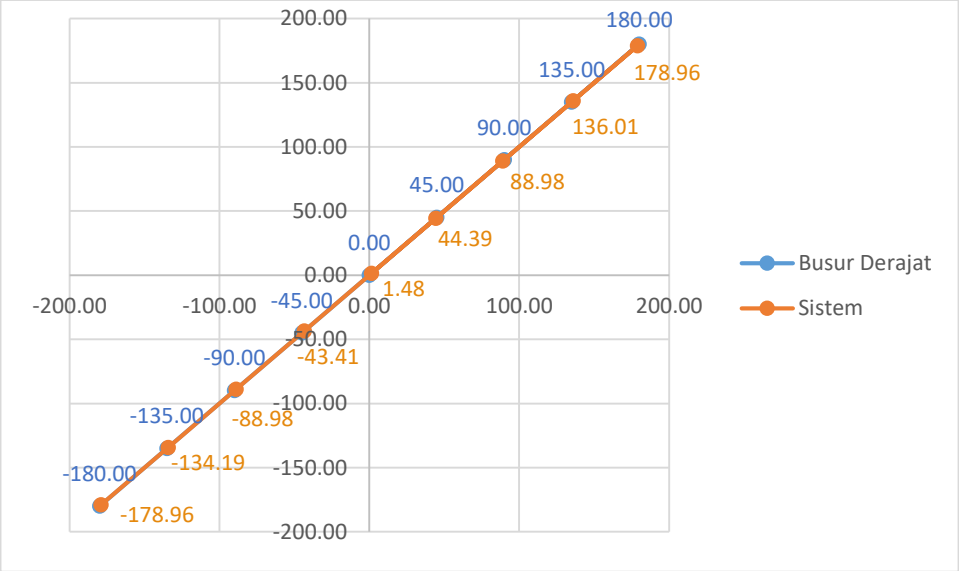
Berikut adalah beberapa contoh tampilan pembacaan sudut pada sistem kamera omnivision saat dilakukan proses pengujian.



(a) **(b)** **(c)**

Gambar 4.13 Contoh Hasil Pembacaan Sudut pada Kamera Omnivision: **(a)**Sisi Kiri (-90°) , **(b)**Sisi Kanan (90°), **(c)**Sisi Belakang (180°/-180°)

Grafik 4.1 menunjukkan perbandingan hasil pembacaan sudut antara sistem dengan busur derajat pada kamera omnivision.



Grafik 4.1 Grafik Perbandingan Hasil Pembacaan Sudut antara Busur Derajat dengan Sistem pada Kamera Omnivision

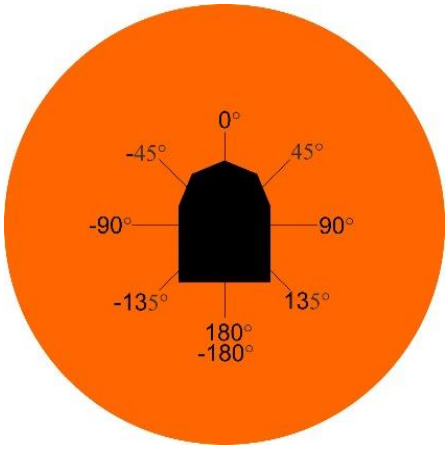
Hasil dari percobaan di atas menunjukkan bahwasannya pengaplikasian algoritma pemberian pemberian sudut pada kamera omnivision dengan menggunakan metode trigonometri segitiga siku-siku telah dapat berjalan dengan baik dan sesuai dengan yang diharapkan, indikator-indikator yang menyatakan hal tersebut antara lain adalah,

1. Ketika bola berada tepat di depan robot, nilai derajat terbaca pada sistem telah mendekati 0°.
2. Ketika bola berada tepat di belakang robot, nilai terbaca pada sistem telah mendekati 180° atau -180°.
3. Ketika bola berada pada zona sisi kanan robot, nilai derajat terbaca pada sistem berhasil menunjukkan nilai positif pada rentang derajat lebih besar dari (>) 0° hingga kurang dari (<) 180°.
4. Sedangkan ketika bola pada zona sisi kiri robot, nilai derajat terbaca pada sistem telah berhasil menunjukkan nilai negatif, yaitu kurang dari (<) 0° hingga lebih besar dari (>) -180°.

Berdasarkan data hasil pengujian sudut pada kamera omnivision, rata-rata nilai selisih pembacaan sudut dari busur derajat dengan sistem adalah sebesar 1,0725◦ atau dengan kata lain memiliki tingkat *error* sebesar 1,0725%. Terdapat beberapa faktor yang memungkinkan menjadi penyebab ketidak akuratan pembacaan nilai sudut dalam pengujian ini, yaitu:

1. Penempatan kamera omnivision yang kurang datar.
2. Penempatan cermin bola yang kurang tepat berada di depan kamera omnivision.
3. *Base* atau landasan tempat meletakkan kamera omnivision, kurang datar dan posisinya sedikit melenceng dari titik tengah robot.
4. Deteksi posisi titik tengah pada objek yang tidak stabil.

Pemberian algoritma pembacaan sudut objek terdeteksi terhadap kamera omnivision akan memberikan banyak bantuan terhadap sistem untuk mengetahui secara lebih akurat posisi dari objek terdeteksi tersebut. **Gambar 4.14** menunjukkan ilustrasi tampilan pembacaan sudut objek terdeteksi terhadap kamera omnivision.



Gambar 4.14 Luas Jangkauan Deteksi Sudut pada Kamera Omnivision

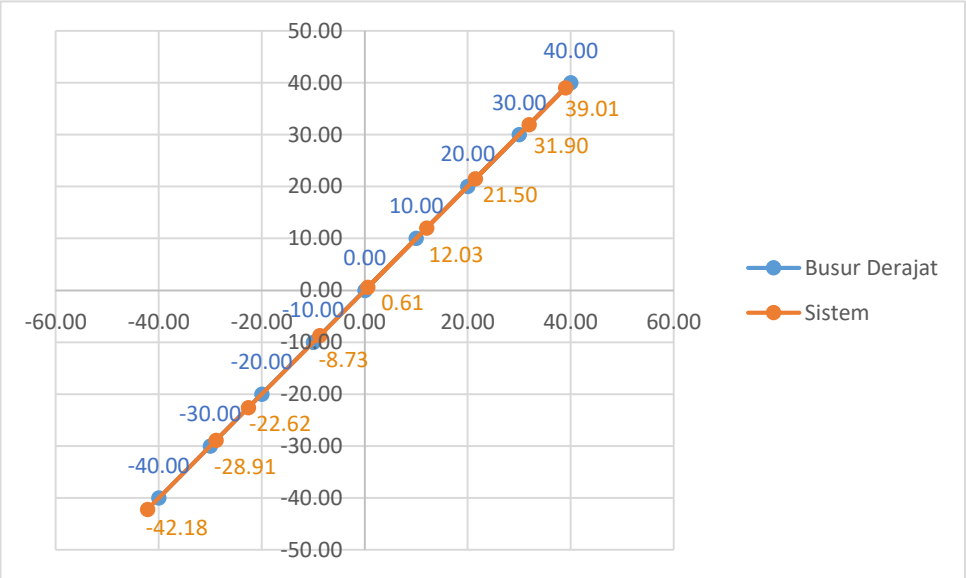
4.3 Pengujian Pembacaan Sudut pada Kamera Depan

Berikut adalah data hasil pengujian pembacaan sudut objek terdeteksi terhadap kamera depan.

Tabel 4.2 Hasil Pengujian Pembacaan Sudut pada Kamera Depan

No	Sudut Objek	Hasil Pembacaan Sudut pada Sistem	Selisih Hasil Pembacaan Sudut	*Error*
1	0°	0,61°	0,61°	0,61%
2	10°	12,03°	2,03°	2,03%
3	20°	21,5°	1,5°	1,5%
4	30°	31,9°	1,9°	1,9%
5	40°	39,01°	0,99°	0,99%
6	-10°	-8,73°	1,27°	1,27%
7	-20°	-22,62°	2,62°	2,62%
8	-30°	-28,91°	1,09°	1,09%
9	-40°	-42,18°	2,18°	2,18%
Rata-Rata Galat			1,58°	1,58%

Grafik 4.2 menunjukkan perbandingan hasil pembacaan sudut antara sistem dengan busur derajat pada kamera depan.



Grafik 4.2 Grafik Perbandingan Hasil Pembacaan Sudut antara Busur Derajat dengan Sistem pada Kamera Depan

Berikut adalah beberapa contoh tampilan pada sistem saat dilakukan pengujian pembacaan sudut objek terdeteksi terhadap kamera depan.

(a) **(b)** **(c)**

Gambar 4.15 Contoh Hasil Pembacaan Sudut pada Kamera Depan: **(a)**Sisi Tengah , **(b)**Sisi Kiri, **(c)**Sisi Kanan *Frame*

Hasil pengujian perbandingan pembacaan sudut antara sistem dengan busur derajat juga menunjukkan hasil yang cukup memuaskan. Berikut adalah Indikator-indikator yang menunjukkan algoritma pemberian sudut pada kamera depan telah dapat berjalan dengan cukup baik,

1. Ketika bola berada tepat di depan robot, nilai sudut terbaca pada sistem telah mendekati 0°.
2. Ketika bola berada disisi kanan *frame*, maka nilai sudut terbaca akan menjadi bernilai positif, yaitu pada *range* derajat lebih dari (>) 0° hingga sudut terjauh objek dapat terdeteksi oleh sisi kanan kamera depan.
3. Ketika bola berada disisi kiri kamera depan, maka nilai sudut terbaca akan menjadi bernilai negatif, yaitu pada *range* derajat kurang dari (<) 0° hingga sudut terjauh objek dapat terdeteksi oleh sisi kiri kamera depan.

Berdasarkan data hasil pengujian di atas, terlihat bahwa rata-rata tingkat nilai *error* pembacaan sudut objek terdeteksi pada kamera depan adalah sebesar 1,58%, atau dengan kata lain rata-rata selisih hasil pembacaan sudut antara sistem dengan busur derajat adalah sebesar 1,58°. Terdapat beberapa faktor yang memungkinkan menjadi penyebab ketidak akuratan pembacaan nilai sudut dalam pengujian ini, yaitu:

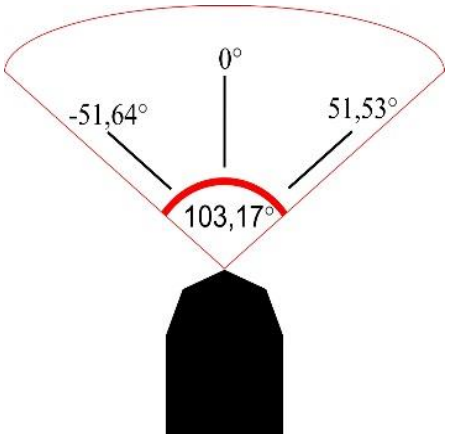
1. Ketidak stabilan deteksi posisi titik tengah dari objek terdeteksi.
2. Posisi penempatan kamera yang tidak tepat berada di tengah-tengah robot.

Berikut adalah data hasil dari pengujian selanjutnya, yaitu pengujian pembacaan sudut terjauh objek dapat terdeteksi pada kamera depan.

Tabel 4.3 Hasil Pengujian Pembacaan Sudut Deteksi Terjauh pada Kamera Depan

No	Jarak Objek	Hasil Pembacaan Sudut				Selisih Hasil Pembacaan Sudut		*Error*	
		pada Sistem		pada Busur Derajat					
		Kanan	Kiri	Kanan	Kiri	Kanan	Kiri	Kanan	Kiri
1	0,3 m	52,01°	-52,32°	50°	-50°	2,01°	2,32°	2,01%	2,32%
2	0,5 m	52,13°	-52,27°			2,13°	2,27°	2,13%	2,27%
3	0,7 m	51,57°	-51,31°			1,57°	1,31	1,57%	1,31%
4	0,9 m	51,11°	-51,02°			1,11°	1,02°	1,11%	1,02%
5	1,1 m	50,52°	-51,34°			0,52°	1,34°	0,52%	1,34%
6	1,3 m	51,47°	-51,57°			1,47°	1,57°	1,47%	1,57%
7	1,5 m	51,92°	-51,65°			1,92°	1,65°	1,92%	1,65%
Rata-Rata		51,53°	-51,64°			1,53°	1,64°	1,53%	1,64%

Gambar 4.16 menunjukkan ilustrasi tampilan luas jangkauan deteksi objek pada kamera depan.



Gambar 4.16 Jangkauan Terjauh Deteksi Objek pada Kamera Depan

Berdasarkan data hasil pengujian pembacaan sudut terjauh objek dapat terdeteksi pada kamera depan, terlihat bahwa luas sudut jangkauan kamera depan

saat mendeteksi suatu objek adalah sebesar 103,17°, dimana nilai tersebut diperoleh dengan menjumlahkan rata-rata hasil terbaca sisi kanan dan kiri pada sistem. Sedangkan berdasarkan pembacaan sudut dari busur derajat, jangkauan luas deteksi objeknya adalah 100°. Dari data di atas terlihat bahwa rata-rata selisih pembacaan sudut pada sistem dengan busur derajat adalah rata-rata 1,53° untuk sisi kanan dan 1,64° untuk sisi kiri kamera, atau dengan kata lain rata-rata tingkat *error*nya adalah 1,53% pada sisi kanan dan 1,64% di sisi kiri.

4.4 Pengujian Deteksi Lapangan

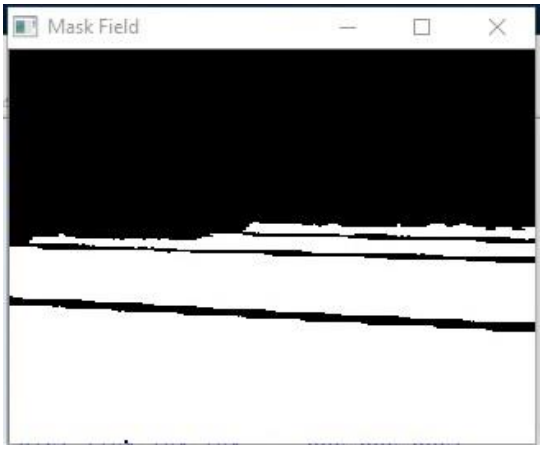
Pada penelitian ini, secara spesifik adalah mendeteksi tepi atau batas dari lapangan sepak bola. Proses deteksi lapangan sepak bola dimulai dari konversi citra RGB ke dalam ruang warna HSV, selanjutnya menentukan nilai parameter batas atas dan bawah warna HSV dari lapangan. Indikator tepat atau tidaknya hasil proses *tuning* warna HSV lapangan akan terlihat jelas pada tahap *masking* atau *thresholding range* warna lapangan. Berikut adalah hasil proses *tuning* nilai parameter batas atas dan bawah warna HSV lapangan,

Tabel 4.4 Hasil Penyetelan Nilai HSV pada Warna Lapangan

No	Parameter	Nilai Parameter Lapangan dalam OpenCV
1	*Hue High*	90
2	*Hue Low*	30
3	*Saturation High*	255
4	*Saturation Low*	100
5	*Value High*	255
6	*Value Low*	100

Setelah dilakukan proses *tuning* warna HSV lapangan, tahap selanjutnya adalah proses *thresholding* atau *masking* warna lapangan. Berdasarkan **Gambar 4.17**, terlihat bahwa proses *masking* warna lapangan telah dapat berjalan dengan baik, yaitu dimana hanya warna lapangan saja yang berwarna putih, sedangkan lainya menjadi hitam. Selain itu juga terlihat garis-garis berwarna hitam yang membentang membelah warna putih, dimana garis-garis hitam tersebut merupakan

garis-garis lapangan. Hasil proses *masking* warna lapangan sekaligus membuktikan bahwa nilai parameter hasil *tuning* warna HSV lapangan sudah tepat.

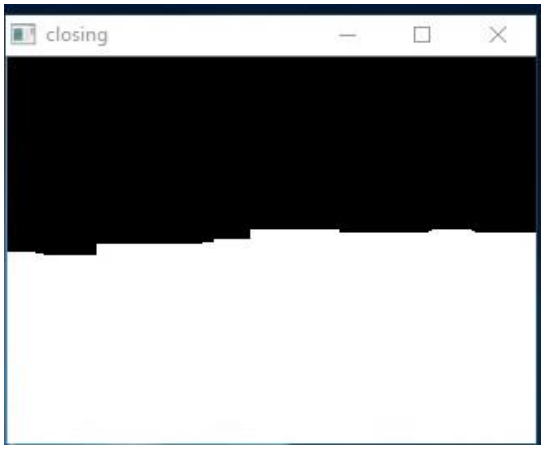


Gambar 4.17 Hasil Proses *Masking* Warna Lapangan

Setelah dilakukan *masking*, selanjutnya adalah tahap *opening* warna lapangan. Terlihat dari hasil pengujian pada **Gambar 4.18**, nilai kernel yang telah disematkan pada proses *opening* deteksi lapangan tidak menghilangkan atau mengeliminasi warna lapangan yang terisolasi oleh garis lapangan. Selain itu, pada tahap ini nilai iterasi antara *erode* dan *dilate* yang digunakan bernilai sama, sehingga dengan demikian keluaran atau *output* dari proses ini tidak akan merubah luas wilayah warna lapangan hasil dari proses *masking* pada tahap sebelumnya. Jadi secara keseluruhan, proses *opening* warna lapangan telah dapat berjalan sesuai dengan yang diharapkan.

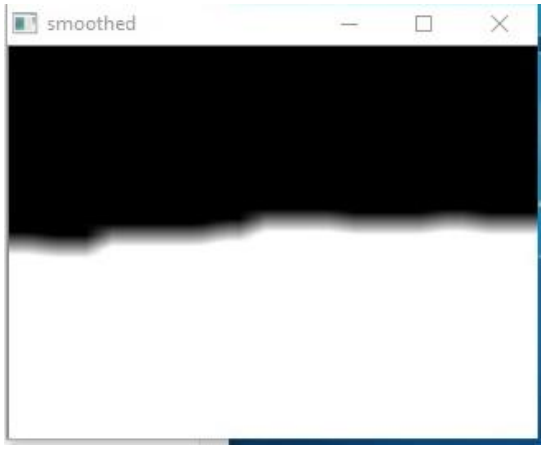
Gambar 4.18 Hasil Proses *Opening* Warna Lapangan

Pada tahap *closing* warna lapangan, nilai kernel yang digunakan adalah matriks berukuran 30 x 30, dimana nilai tersebut terbilang cukup besar, sehingga dipastikan cukup untuk menutupi sekat-sekat yang ditimbulkan oleh garis lapangan. Selain itu, nilai iterasi *dilate* dan *erode* yang digunakan di dalam proses ini adalah sama, sehingga hasil dari tahap ini tidak merubah luas wilayah warna lapangan. Berdasarkan hasil pengujian yang telah dilakukan, Tahap *closing* warna lapangan telah dapat berjalan dengan baik. Hal tersebut terlihat dari warna lapangan yang sebelumnya terpisah atau tersekat karena keberadaan garis lapangan, setelah melalui proses *closing* dapat menyatu menjadi satu lagi, atau dengan kata lain tidak lagi tersekat oleh garis lapangan. Jadi dengan demikian, fungsi *closing* yang bertujuan untuk menutup *noise-noise* warna lain yang terdapat di antara warna lapangan, telah dapat tercapai dengan baik. Berikut adalah tampilan hasil proses *closing* warna lapangan.



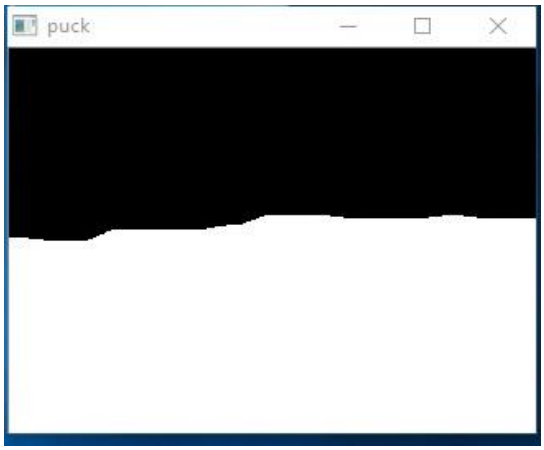
Gambar 4.19 Hasil Proses *Closing* Warna Lapangan

Terlihat pada **Gambar 4.20**, hasil pengujian citra keluaran dari proses 2D *Convolution* berhasil menjadi tersamarkan atau *blur*. Indikator tersebut membuktikan bahwa proses filter ini telah dapat berjalan dengan baik. Hasil citra *blur* tersebut mengakibatkan luas wilayah warna lapangan menjadi lebih luas daripada sebelumnya.



Gambar 4.20 Hasil Proses Filter *2D Convolution*Warna Lapangan

Hasil dari tahap *thresholding* akan membuktikan apakah tujuan dari proses filter 2D *Convolution* dapat tercapai ataukah tidak. Dari hasil pengujian, terlihat bahwa *output* dari Tahap *thresholding* telah berhasil mengembalikan citra *blur* menjadi jelas kembali. Terlihat pada **Gambar 4.21**, daerah warna putih (warna lapangan) menjadi sedikit lebih luas daripada keluaran dari proses *closing*, hal tersebut membuktikan bahwa tujuan dari proses filter 2D *Convolution* telah tercapai.



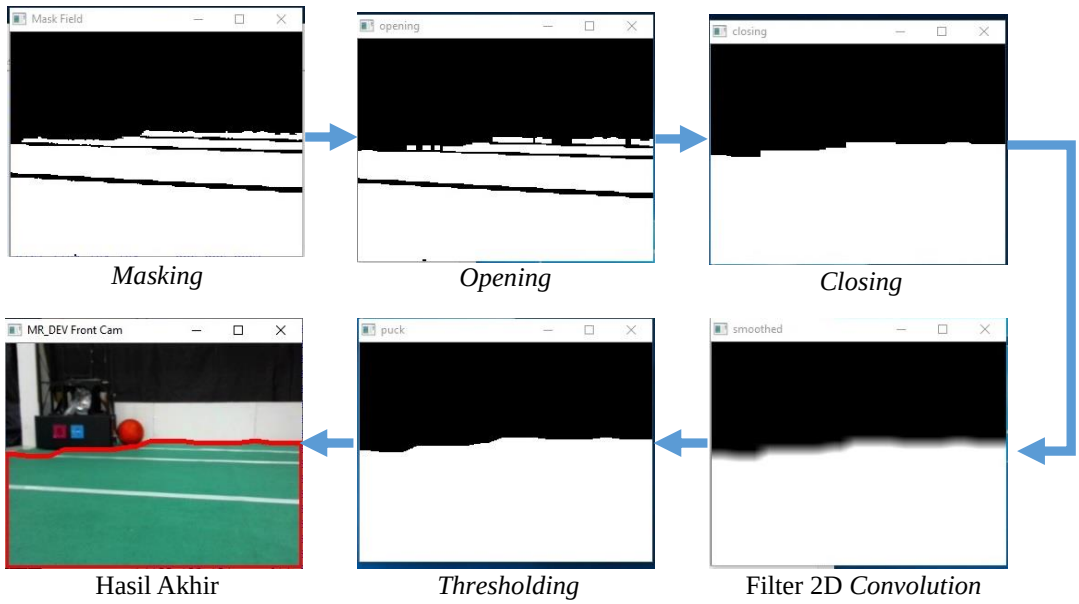
Gambar 4.21 Hasil Proses *Thresholding* Warna Lapangan

Tahap terakhir dari proses deteksi lapangan adalah menemukan kontur warna lapangan, dengan mengetahui kontur warna lapangan, itu artinya akan mendapatkan data batas-batas dari wilayah warna lapangan. Pada **Gambar 4.22** terlihat bahwa kontur warna lapangan telah berhasil didapatkan, dan kemudian

digambarkan dengan garis berwarna merah. Hal tersebut merupakan indikator bahwa proses pencarian kontur lapangan telah berhasil dilakukan, sehingga dengan demikian data batas-batas wilayah warna lapangan tersebut selanjutnya dapat digunakan dalam proses anti-*noise* warna bola.

Gambar 4.22 Hasil Proses Kontur Warna Lapangan

Berikut adalah tampilan secara keseluruhan hasil dari setiap tahapan proses pendeteksian tepi atau batas lapangan sepak bola.



Gambar 4.23 Tahapan Proses Deteksi Tepi Lapangan Sepak Bola

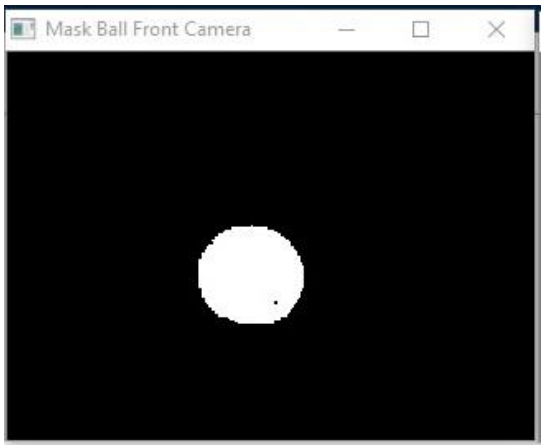
4.5 Pengujian Deteksi Bola pada Kamera Depan

Proses deteksi bola pada kamera depan diawali dengan mengkonversi citra RGB ke dalam ruang warna HSV. Kemudian melakukan proses penyetelan nilai parameter batas atas dan bawah warna HSV bola pada kamera depan. Berikut adalah nilai parameter hasil proses *tuning* warna HSV bola yang digunakan,

Tabel 4.5 Hasil Penyetelan Nilai HSV Warna Bola pada Kamera Depan

No	Parameter	Nilai Parameter Lapangan dalam OpenCV
1	*Hue High*	20
2	*Hue Low*	0
3	*Saturation High*	255
4	*Saturation Low*	199
5	*Value High*	255
6	*Value Low*	91

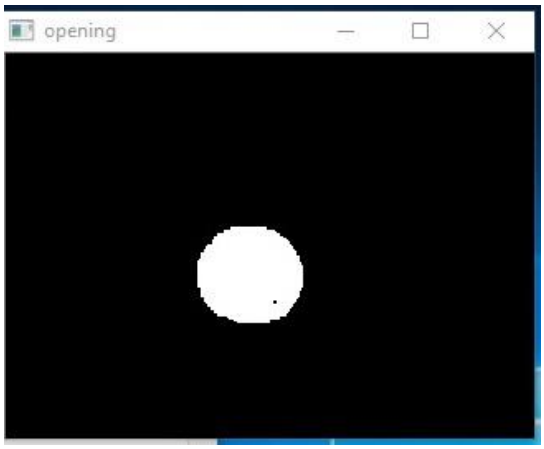
Terlihat dari hasil proses *masking* pada **Gambar 4.24**, warna yang menjadi warna bola berhasil berubah menjadi berwarna putih, diluar *range* warna HSV yang telah di*setting*, maka akan berubah menjadi berwarna hitam. Hal tersebut menunjukkan bahwa nilai parameter hasil proses *tuning* nilai parameter batas atas dan bawah warna HSV bola pada kamera depan telah tepat.



Gambar 4.24 Hasil Proses *Masking* Warna Bola pada Kamera Depan

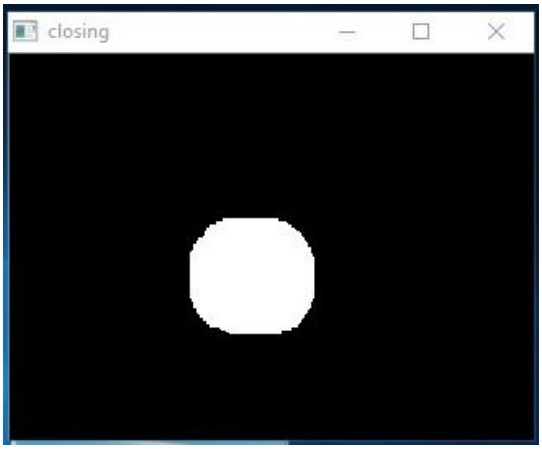
Tahap selanjutnya adalah proses *opening* warna bola pada kamera depan. Ukuran kernel yang digunakan sama seperti pada proses deteksi lapangan. Selain

itu, nilai iterasi antara *erode* dan *dilate* yang digunakan pun juga sama. Sehingga, luas wilayah warna bola tetap berukuran sama seperti hasil keluaran dari tahap *masking*. Hal tersebut sengaja dilakukan agar proses *opening* tidak mengkorosi luas wilayah warna bola. Hal tersebut sangat penting karena apabila luas wilayah warna bola berkurang, maka akan berdampak pada berkurangnya jarak maksimum deteksi bola pada kamera depan. Berdasarkan hasil pengujian, proses *opening* warna bola pada kamera depan telah dapat berjalan dengan baik seperti apa yang telah dijelaskan sebelumnya. Jadi dengan demikian, fungsi tahap ini sebagai peminimalisir *noise* tetap dapat berjalan dengan baik tanpa harus mengorbankan jarak maksimum deteksi bola.



Gambar 4.25 Hasil Proses *Opening* Warna Bola pada Kamera Depan

Pada tahap *closing* warna bola, nilai kernel yang digunakan adalah matriks berukuran 5 x 5. Sedangkan iterasi *dilate* bernilai 5 kali dan *erode* bernilai 2 kali. Berdasarkan hasil dari pengujian, titik-titik hitam yang tadinya ada di antara warna putih sudah tidak lagi terlihat. Jadi dengan kata lain, *noise-noise* warna lain yang menempel pada bola telah berhasil ditutupi. Selain itu, luas wilayah warna bola pada kamera depan juga menjadi sedikit lebih besar, hal itu dikarenakan nilai iterasi *dilate* lebih besar daripada *erode*. Tujuan dari dilakukannya hal tersebut adalah agar bola dapat dideteksi dari jarak yang cukup jauh.



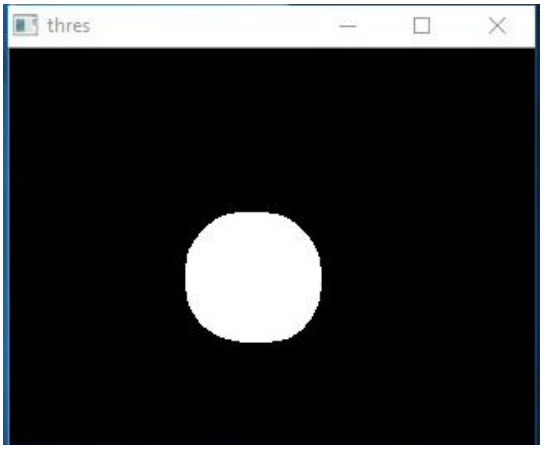
Gambar 4.26 Hasil Proses *Closing* Warna Bola pada Kamera Depan

Kemudian, tahap yang dilalui adalah proses filter 2D Convolution. Terlihat dari **Gambar 4.27**, citra keluaran dari proses 2D *Convolution* berhasil menjadi tersamarkan atau *blur*. Indikator tersebut menunjukkan bahwa proses filter warna bola pada kamera depan telah berjalan dengan baik.

Gambar 4.27 Hasil Proses Filter 2D *Convolution* Warna Bola pada Kamera Depan

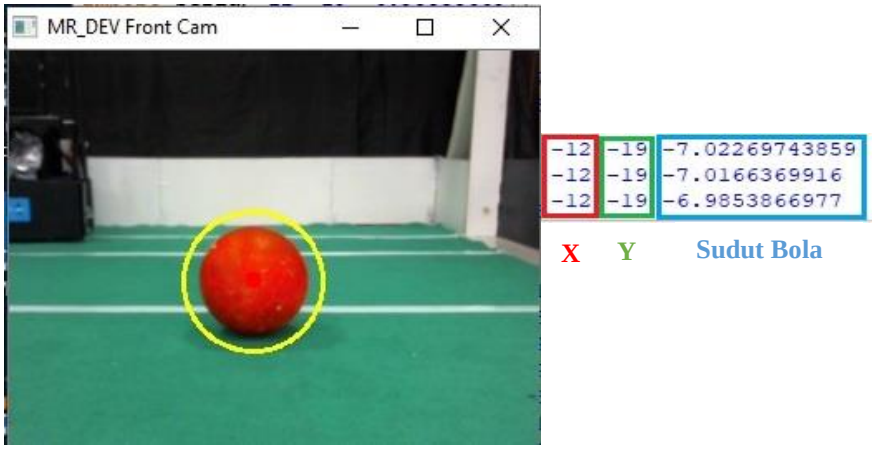
Tahap selanjutnya adalah mengembalikan citra *blur* kembali menjadi tajam, yaitu dengan menggunakan proses *thresholding*. Berdasarkan hasil pengujian, proses *thresholding* telah berhasil mengembalikan citra yang tersamarkan menjadi jelas kembali. Selain itu, terlihat dari hasil keluarannya, luas wilayah warna bola menjadi lebih luas dibandingkan dengan keluaran pada proses *closing*. Indikator tersebut sekaligus membuktikan bahwa tujuan dari proses filter 2D *Convolution*,

telah tercapai sesuai dengan yang diharapkan. Jadi dengan demikian, bola dapat dideteksi dari jarak yang cukup jauh.



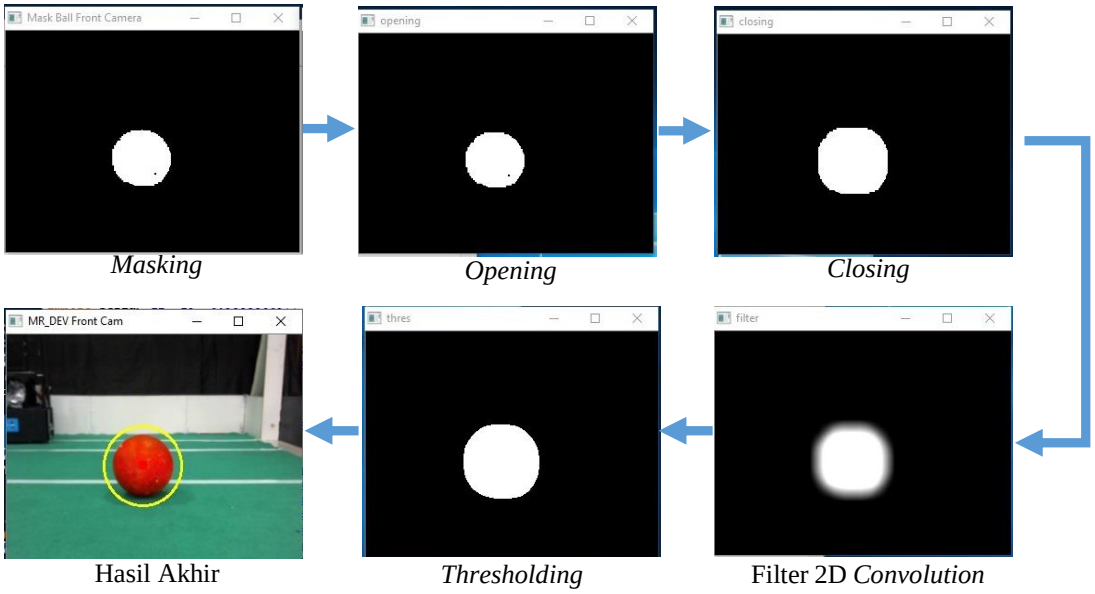
Gambar 4.28 Hasil Proses *Thresholding* Warna Bola Kamera Depan

Tahap terakhir proses deteksi bola pada kamera depan adalah menemukan kontur warna bola, sehingga kemudian dapat diperoleh titik koordinatnya dengan menggunakan metode analisis *moment*.



Gambar 4.29 Hasil Akhir Deteksi Bola pada Kamera Depan

Berikut adalah tampilan hasil secara keseluruhan dari tahapan proses pendeteksian bola pada kamera depan.



Gambar 4.30 Tahapan Proses Deteksi Bola pada Kamera Depan

hasil dari pengujian, proses ini telah dapat berjalan dengan baik, indikator-indikator yang menyatakan keberhasilan tersebut, antara lain adalah,

1. Ketika bola berada tepat di tengah *frame*, nilai koordinat *x* bola telah berhasil mendekati nol (0).
2. Pada saat bola berada di sisi kanan *frame*, nilai koordinat *x* bola telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 159.
3. Sebaliknya, saat bola berada di sisi kiri *frame*, nilai koordinat *x* bola telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -159.
4. Ketika bola berada tepat di tengah *frame*, nilai koordinat *y* bola telah berhasil mendekati nol (0).
5. Pada saat bola berada di sisi atas *frame*, nilai koordinat *y* bola telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 119.
6. Sebaliknya, saat bola berada di sisi bawah *frame*, nilai koordinat *y* bola telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -119.

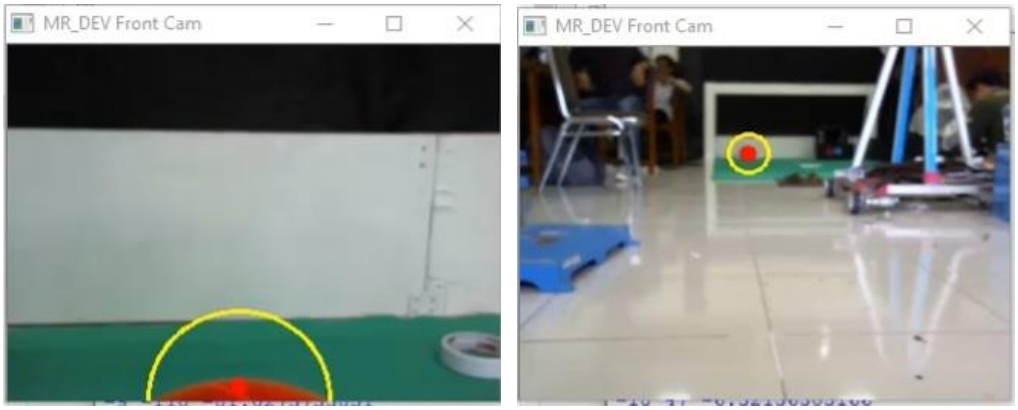
Pengujian selanjutnya adalah menguji jarak minimum dan maksimum bola dapat terdeteksi oleh kamera depan, berikut adalah hasil pengujian yang didapatkan,

Tabel 4.6 Jarak Deteksi Minimum dan Maksimum Bola pada Kamera Depan

No	Posisi Bola	Jarak	
		Minimum Deteksi	Maksimum Deteksi
1	0°	0,2 m	>7,2 m
2	10°	0,2 m	>7,2 m
3	20°	0,2 m	>7,2 m
4	30°	0,2 m	>7,2 m
5	40°	0,2 m	>7,2 m

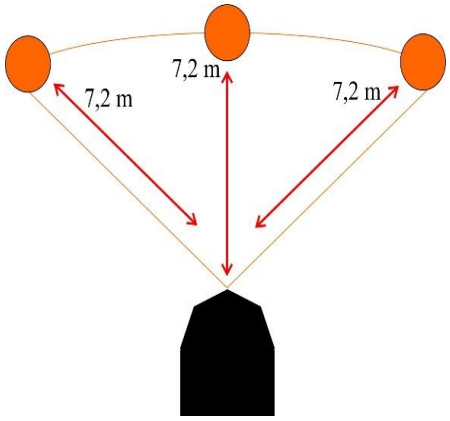
Terlihat dari data hasil pengujian pada tabel di atas, bahwa ketika posisi bola tepat berada di depan kamera atau 0°, maka jarak minimum bola terhadap kamera depan agar dapat terdeteksi adalah 0,2 m, dan jarak deteksi terjauhnya adalah lebih dari 7,2 m. Dikatakan lebih dari 7,2m karena pada jarak tersebut proses deteksi bola pada kamera depan masih berjalan lancar dan pengukuran jarak maksimal terpaksa berhenti sampai pada titik tersebut dikarenakan keterbatasan ruangan tempat pengujian. Hal tersebut juga terjadi pada pengujian posisi bola yang lainnya, pengujian terpaksa dihentikan dikarenakan lebar tempat pengujian yang terbatas.

Hasil pengukuran jarak tersebut terbilang sukses dikarenakan sistem mampu mendeteksi bola dengan jarak melebihi panjang setengah lapangan yang digunakan dalam KRSBI-Beroda, yaitu 4,5 m. Itu artinya pada proses *kick off,* robot akan dapat mendeteksi bola yang berada di tengah lapangan dengan mudah.



Gambar 4.31 (a) Jarak Minimum dan **(b)**Jarak Maksimum Deteksi Bola Kamera Depan

Gambar 4.32 menunjukkan ilustrasi jarak jangkauan deteksi bola pada kamera depan.



Gambar 4.32 Ilustrasi Jarak Jangkuan Deteksi Bola pada Kamera Depan

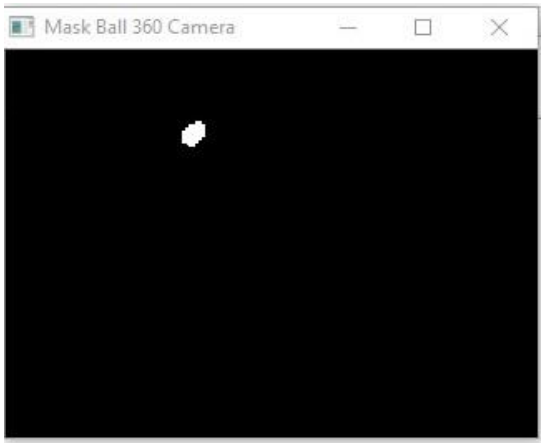
4.6 Pengujian Deteksi Bola pada Kamera Omnivision

Nilai parameter HSV warna bola pada kamera omnivision mirip seperti pada kamera depan, yang membedakan adalah sedikit penyesuaian pada bagian *saturation* dan *value.* Hal tersebut dikarenakan pada kamera omnivision, kamera tidak langsung menghadap ke arah objek, melainkan membaca pantulan dari cermin bola, selain itu juga terdapat *cover* tabung akrilik yang mengitarinya, sehingga mempengaruhi nilai intensitas cahaya yang masuk ke dalam kamera. Jadi dengan demikian, perlu diadakan penyesuaian pada bagian *saturation* yang mana mengatur tentang tingkat kekuatan warna (muda atau tua), dan bagian *value* yang mengatur tentang tingkat pengaruh intensitas cahaya terhadap warna. Berikut adalah data nilai-nilai parameter hasil dari proses *tunning* warna HSV.

Tabel 4.7 Hasil Penyetelan Nilai HSV Warna Bola pada Kamera Omnivision

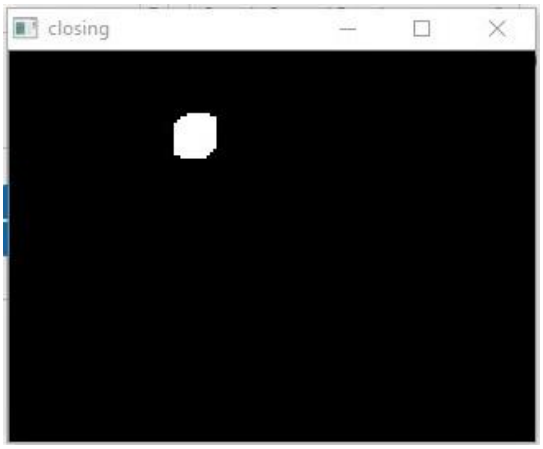
No	Parameter	Nilai Parameter Lapangan dalam OpenCV
1	*Hue High*	20
2	*Hue Low*	0
3	*Saturation High*	255
4	*Saturation Low*	195
5	*Value High*	255
6	*Value Low*	49

Hasil dari proses *tunning* nilai parameter warna bola pada kamera omnivision telah tepat. Hal tersebut terlihat dari hasil proses *masking*nya, dimana objek warna bola berubah menjadi berwarna putih, sedangkan warna-warna lain yang berada di luar nilai parameter batas atas dan bawah warna HSV bola akan berubah menjadi berwarna hitam.



Gambar 4.33 Hasil Proses *Masking* Warna Bola Kamera Omnivision

Pada deteksi bola kamera omnivision tidak melewati proses *opening* terlebih dahulu, karena dikhawatirkan justru akan menghilangkan hasil dari proses *masking.* Hal tersebut dikarenakan ukuran luas wilayah warna bola keluaran dari proses *masking* pada kamera omnivision sangatlah kecil. Jadi dengan demikian, proses *opening* langsung dilewati ke tahap *closing* untuk menutupi bercak-bercak hitam yang terdapat diantara hasil *masking* warna bola. Nilai kernel yang digunakan dalam proses *closing* warna bola adalah matriks berukuran 5 x 5. Sedangkan nilai iterasi *dilate* yang digunakan adalah 5 kali, dan *erode* 2 kali. Terlihat dari **Gambar 4.34**, warna bola pada kamera omnivision telah dapat tertutupi dengan sempurna. Selain itu, luas wilayah warna bola juga bertambah dikarenakan nilai iterasi *dilate* lebih besar daripada *erode.* Hal tersebut dilakukan dengan tujuan untuk menambah jarak maksimum deteksi bola pada kamera omnivision.

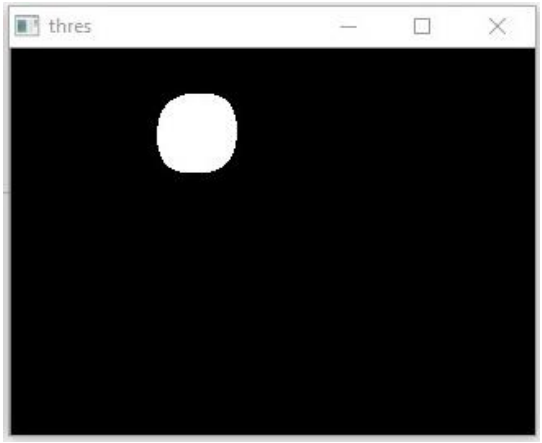


Gambar 4.34 Hasil Proses *Closing* Warna Bola Kamera Omnivision

Terlihat pada **Gambar 4.35**, citra keluaran dari proses 2D *Convolution* berhasil menjadi tersamarkan atau *blur*. Indikator tersebut menunjukkan bahwa proses filter warna bola pada kamera depan telah berjalan dengan baik.

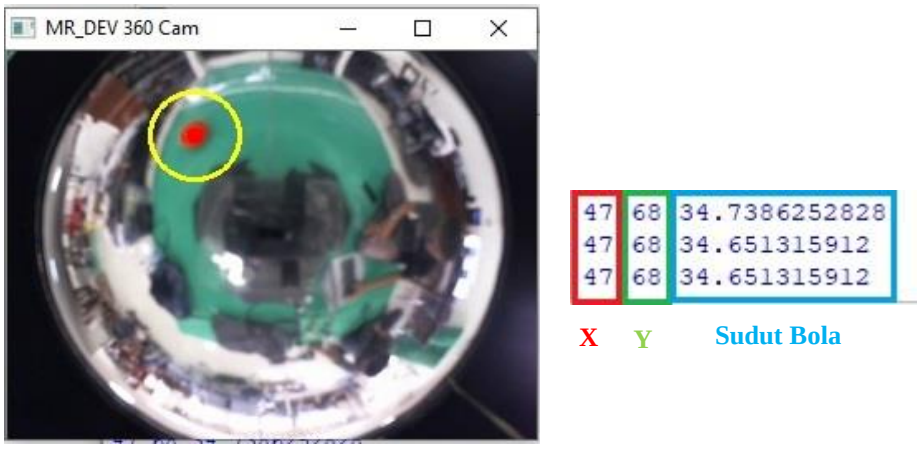
Gambar 4.35 Hasil Proses Filter *Smoothed* Warna Bola Kamera Omnivision

Berdasarkan hasil pengujian pada **Gambar 4.36**, terlihat bahwa proses *thresholding* telah dapat berjalan dengan baik. Hal tersebut terlihat dari hasil citra keluarannya yang kembali menjadi jelas. Selain itu, tujuan dari filter 2D *Convolution* pun juga berhasil, hal tersebut terbukti dari luas wilayah warna bola pada kamera omnivision telah bertambah besar daripada sebelumnya. Jadi dengan demikian akan lebih meningkatkan jarak maksimum pendeteksian bola pada kamera omnivision.



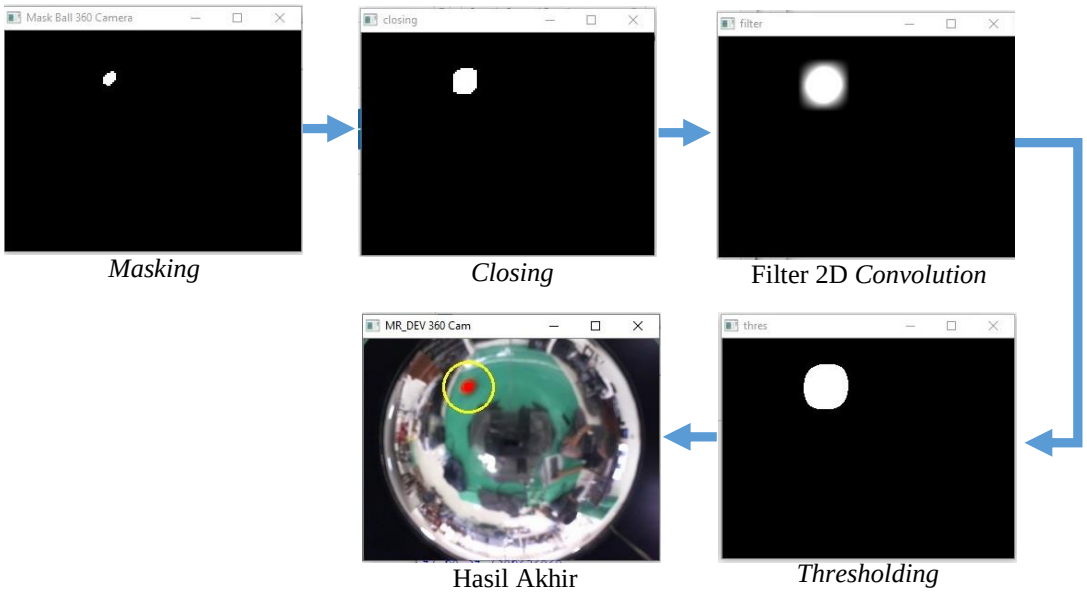
Gambar 4.36 Hasil Proses *Thresholding* Warna Bola Kamera Omnivision

Tahap terakhir adalah menemukan kontur warna bola pada kamera omnivision. Setelah kontur dapat ditemukan, maka kemudian titik koordinat bola pada kamera omnivision dapat ditemukan dengan menggunakan metode analisis *moment*. Metode tersebut memungkinkan untuk dapat menemukan titik pusat grafitasi dari warna yang terdeteksi. Jadi dengan demikian dapat mendapat data titik koordinat (x, y) dari titik pusat grafitasi warna bola pada kamera omnivision. Titik koordinat tersebutlah yang kemudian dimanfaatkan sebagai lokasi koordinat dari bola.



Gambar 4.37 Hasil Akhri Deteksi Bola pada Kamera Omnivision

Berikut adalah tampilan secara keseluruhan hasil dari seluruh tahapan proses pendeteksian bola pada kamera omnivision.



Gambar 4.38 Tahapan Proses Deteksi Bola pada Kamera Omnivision

Berikut adalah indikator-indikator yang menunjukkan bahwa proses pendeteksian bola pada kamera omnivision telah dapat berjalan dengan baik.

1. Ketika bola berada tepat di tengah *frame*, nilai koordinat *x* bola telah berhasil mendekati nol (0).
2. Pada saat bola berada di sisi kanan *frame*, nilai koordinat *x* bola telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 159.
3. Sebaliknya, jika bola berada di sisi kiri *frame*, nilai koordinat *x* bola juga telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -159.
4. Ketika bola berada tepat di tengah *frame*, nilai koordinat *y* bola telah berhasil mendekati nol (0).
5. Pada saat bola berada di sisi atas *frame*, nilai koordinat *y* bola telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 119.
6. Sebaliknya, saat bola berada di sisi bawah *frame*, nilai koordinat *y* bola telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -119.

Pengujian selanjutnya adalah menguji jarak minimum dan maksimum bola dapat terdeteksi pada kamera omnivision. Berikut adalah data hasil pengujian yang telah dilakukan,

Tabel 4.8 Hasil Pengujian Jarak Minimum dan Maksimum Deteksi Bola pada Kamera Omnivision

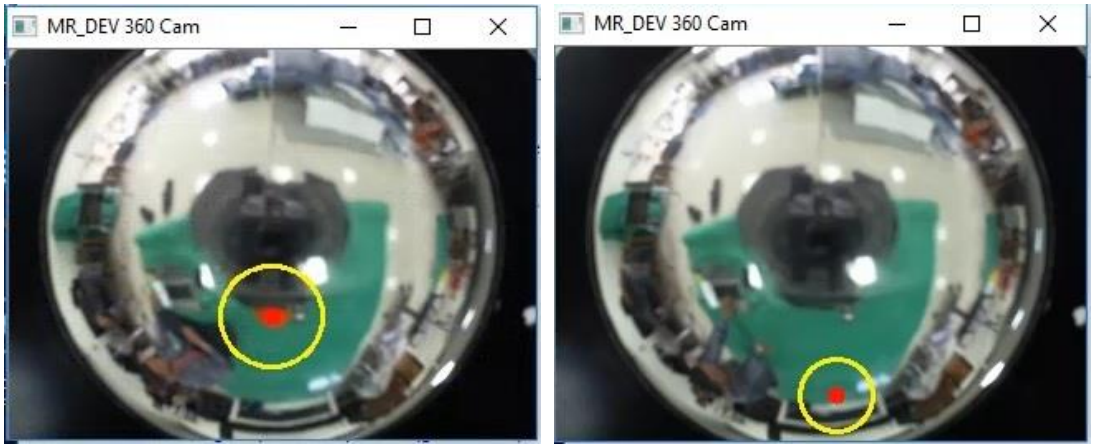
No	Posisi Bola	Jarak	
		Minimum Deteksi	Maksimum Deteksi
1	0°	Menempel Robot	1,65 m
2	45°	Menempel Robot	1,94 m
3	90°	Menempel Robot	1,91 m
4	135°	Menempel Robot	1,97 m
5	180°/-180°	Menempel Robot	1,83 m
6	-45°	Menempel Robot	1,87 m
7	-90°	Menempel Robot	1,93 m
8	-135°	Menempel Robot	1,89 m
Rata-Rata Jarak Terjauh			1,87 m

Terlihat pada tabel di atas, bahwasannya sistem deteksi bola pada kamera omnivision telah berhasil mendeteksi keberadaan bola pada saat berada di titik buta kamera depan. Hal tersebut mengindikasikan tujuan awal ditambahkannya kamera omnivision telah tercapai, yaitu untuk menutupi kekurangan sudut pandang dari kamera depan yang relatif sempit.

Dari data tersebut, diketahui bahwa sistem deteksi bola pada kamera omnivision berhasil mendeteksi bola pada posisi bola menempel dengan *body* robot pada semua sudut (360°). Hal tersebut merupakan hal yang positif, yaitu dimana ketika suatu ketika bola berada di titik buta kamera depan dan menempel dengan *body* robot, maka akan tetap dapat terdeteksi.

Terlihat bahwa pada saat deteksi bola tepat di depan robot (0°), jangkauan jarak maksimum deteksinya terlihat lebih rendah dibandingkan dengan sudut-sudut lainnya. Hal tersebut dikarenakan tepat pada sisi tersebut, terdapat sambungan tabung akrilik yang melingkupi kamera omnivision, sehingga sedikit mengganggu pandangan kamera pada sisi tersebut. Posisi sambungan tabung akrilik tersebut sengaja diletakkan tepat di sisi depan robot dikarenakan pada sisi tersebut termasuk kedalam jangkauan deteksi dari kamera depan. Jadi dengan kata lain, pada sisi tersebut masih ter*backup* oleh kamera depan, sehingga tidak menjadi sebuah masalah.Berdasarkan data tersebut, rata-rata jangkuan jarak deteksi bola terjauh

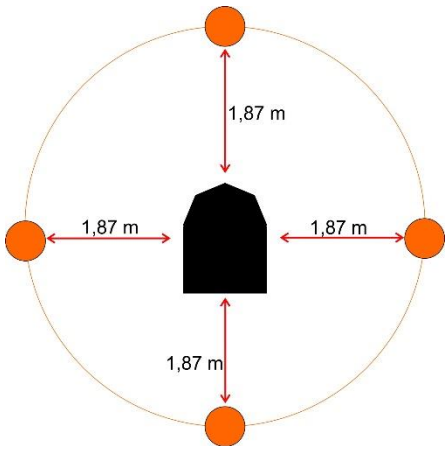
pada kamera omnivision di semua sisinya adalah 1,87 m. Hal tersebut tentu akan sangat membantu memperluas area jangkauan deteksi bola pada robot Mr Dev.



(a) **(b)**

Gambar 4.39 Contoh Pengujian Jarak Deteksi **(a)**Minimum dan **(b)**Maksimum pada Kamera Omnivision

Gambar 4.40 menunjukkan ilustrasi jangkauan terjauh deteksi bola pada kamera omnivision.



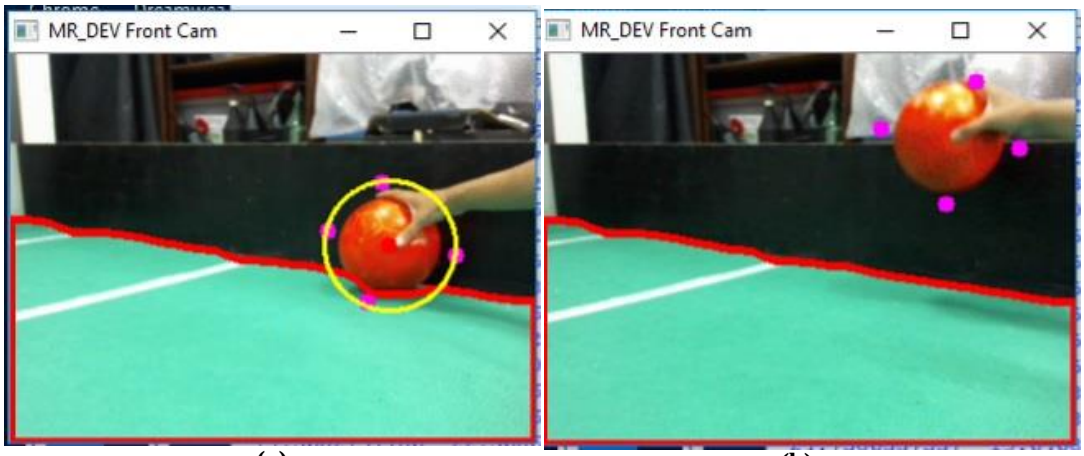
Gambar 4.40 Ilustrasi jangkauan terjauh deteksi Bola pada Kamera Omnivision

4.7 Pengujian Anti-*Noise* Warna Bola

Sistem anti *noise* bola bertujuan agar sistem tidak mendeteksi warna serupa bola yang berada di luar arena lapangan sepak bola, sehingga dengan demikian robot tidak akan mengejar objek *noise* tersebut. Oleh karena itulah maka akan diaplikasikan baik pada kamera depan maupun pada kamera omnivision. Pengujian

akan dilakukan dua kali, yaitu pengujian sistem anti *noise* bola pada kamera depan dan pada kamera omnivision.

Pengujian pertama adalah menguji sistem anti *noise* bola pada kamera depan. Berdasarkan hasil pengujian pada kamera depan, bola mulai tidak terdeteksi oleh sistem ketika diangkat di atas ketinggian 8,5 cm. Hal tersebut merupakan hasil yang positif, dimana itu artinya jika terdapat warna serupa bola yang berada di luar pembatas, maka tidak akan dapat terdeteksi oleh sistem.



(a) **(b)**

Gambar 4.41 Hasil Pengujian Anti-*Noise* Warna Bola pada kamera Depan: **(a)**Di Dalam Lapangan **(b)**Di Luar Lapangan

Pengujian kedua adalah menguji sistem anti *noise* bola pada kamera omnivision. Hasil pengujian pada kamera omnivision, menunjukkan bahwa bola harus diangkat setinggi 29 cm agar bola tidak lagi dapat terdeteksi oleh sistem. Nilai ketinggian tersebut masih dibawah tinggi dari papan pembatas yang digunakan, sehingga hasil pengujian pada kamera omnivision juga tergolong berhasil. Jadi dengan kata lain apabila terdapat objek memiliki warna serupa dengan bola yang berada di luar lapangan maka tidak akan terdeteksi oleh sistem.

(a) **(b)**

Gambar 4.42 Hasil Pengujian Anti-*Noise* Warna Bola pada kamera Omnivision:
(a) Di Dalam Lapangan **(b)**Di Luar Lapangan

Sistem anti *noise* warna bola sukses berjalan dengan baik, hal tersebut membuktikan bahwa algoritma pemberian titik-titik ekstrim pada bola yang dikombinasikan dengan *point polygon test* dapat teraplikasikan dengan baik dalam mengurangi tingkat *noise* pada sistem deteksi bola. Bola akan tetap terdeteksi selama salah satu dari 4 titik ekstrim bola masih terdapat di dalam wilayah kontur lapangan. 4 titik ekstrim bola tersebut antara lain adalah titik teratas, terbawah, terkanan dan terkiri dari bola. Ketika titik tersebut berada di dalam kontur dari lapangan, maka nilai jarak titik tersebut terhadap kontur terdekat lapangan adalah bernilai positif dan saat berada tepat pada garis kontur akan bernilai nol (0). Sedangkan saat titik tersebut berada diluar kontur lapangan, maka nilai jarak titik tersebut terhadap garis kontur terdekat akan bernilai negatif. Jadi dengan demikian, ketika ada warna serupa bola yang berada di luar lapangan, maka sistem tidak akan mendeteksi *noise* tersebut.

4.8 Pengujian Deteksi Tanda Warna Lawan

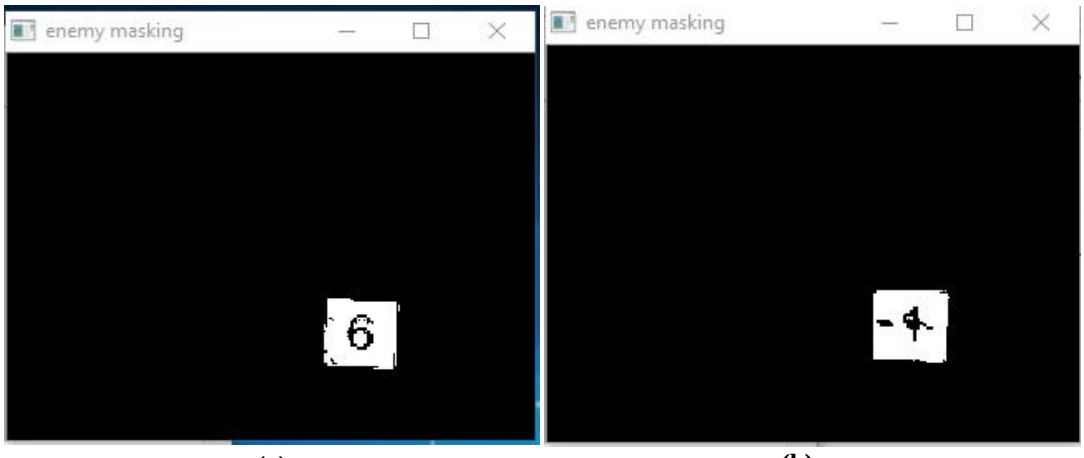
Seperti yang telah dibahas pada pengujian fitur pergantian tanda warna lawan, perintah tersebut dapat dengan mudah diakses dengan mengetikkan karakter “m” untuk magenta dan “c” untuk cyan. Kedua warna tersebut melalui alur proses pendeteksian yang sama seperti deteksi bola pada kamera omnivision.

Tahap pertama yang dilalui adalah mengkonversi citra RGB ke dalam ruang warna HSV, kemudian menyetel nilai *range* HSV dari kedua warna tersebut. Berikut adalah hasil proses *tunning* warna HSV tanda warna lawan.

Tabel 4.9 Hasil Penyetelan Nilai HSV Warna Lawan

No	Parameter	Nilai Parameter	
		Magenta	Cyan
1	*Hue High*	179	110
2	*Hue Low*	140	90
3	*Saturation High*	255	255
4	*Saturation Low*	150	120
5	*Value High*	255	255
6	*Value Low*	70	120

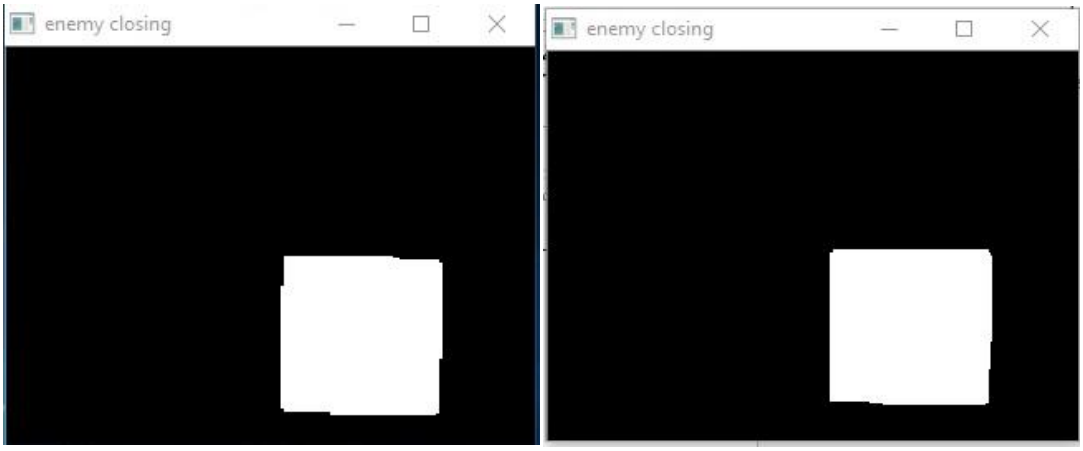
Proses selanjutnya adalah *masking range* warna HSV tanda warna lawan dengan menggunakan nilai parameter hasil proses *tuning* warna HSV yang telah didapatkan. Terlihat pada **Gambar 4.43**, proses *masking* dapat berjalan dengan baik. Semua warna yang merupakan tanda warna lawan berubah menjadi putih, dan selain warna tersebut berubah menjadi hitam. Hal tersebut sekaligus membuktikan bahwa nilai parameter hasil proses *tunning* sudah tepat.



(a) **(b)**

Gambar 4.43 Hasil Pengujian Proses *Masking* pada Deteksi Tanda Warna Lawan: **(a)**Magenta, **(b)**Cyan

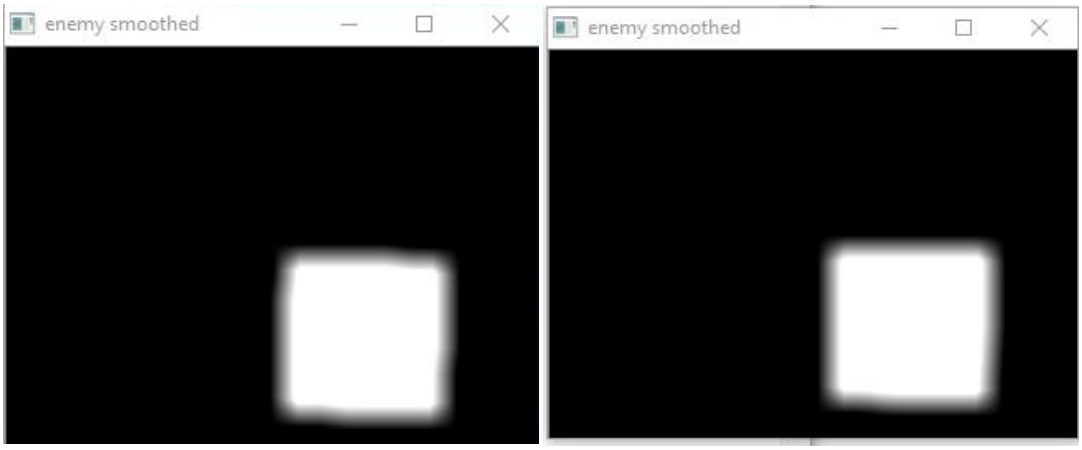
Seperti halnya pada deteksi bola kamera omnivision, proses deteksi tanda warna lawan juga tidak melalui tahap *opening*,melainkan langsung menuju pada tahap *closing.* Hal tersebut dikarenakan ukuran luas wilayah tanda warna lawan hasil keluaran *masking* sangatlah kecil, sehingga dikhawatirkan akan terkikis habis ketika melalui proses *opening.* Pada tahap proses *closing* ini, nilai kernel yang digunakan adalah matriks berukuran 5 x 5. Selain itu, nilai iterasi *dilate* yang digunakan adalah 15 kali, sedangkan *erode* hanya 2 kali saja. Berdasarkan hasil pengujian yang telah dilakukan, proses *closing* telah dapat berjalan dengan baik. Indikator yang menunjukkan hal tersebut adalah angka berwarna hitam di tengah tanda warna lawan telah menghilang sepenuhnya berubah menjadi putih seutuhnya. Itu artinya proses *closing* telah berhasil menutup semua lubang yang ada diantara tanda warna lawan. Selain itu, terlihat bahwa luas wilayah warna pada *output* citra dari proses ini terlihat lebih besar dari sebelumnya, hal tersebut dikarenakan nilai iterasi *dilate* yang lebih besar daripada *erode*. Bertambahnya luas wilayah warna tersebut akan sangat membantu sistem untuk dapat mendeteksi tanda warna lawan dalam jarak yang lebih jauh.



(a) **(b)**

Gambar 4.44 Hasil Pengujian Proses *Closing* pada Deteksi Tanda Warna Lawan: **(a)**Magenta **(b)**Cyan

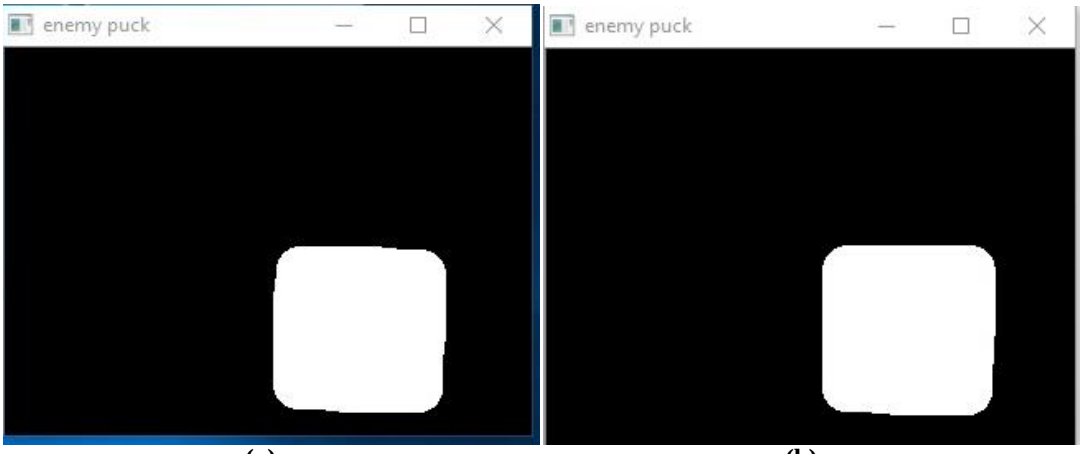
Tahap yang dilalui setelah *closing*, adalah tahap proses filter citra dengan menggunakan 2D *Convolution.* Terlihat dari **Gambar 4.45**, citra hasil keluaran proses filter 2D *Convolution* berubah menjadi tersamarkan atau *blur*. Hal tersebut membuktikan bahwa tahap ini telah dapat terlaksana dengan baik.



(a) **(b)**

Gambar 4.45 Hasil Pengujian Proses Filter 2D *Convolution* pada Deteksi Tanda Warna Lawan: **(a)**Magenta **(b)**Cyan

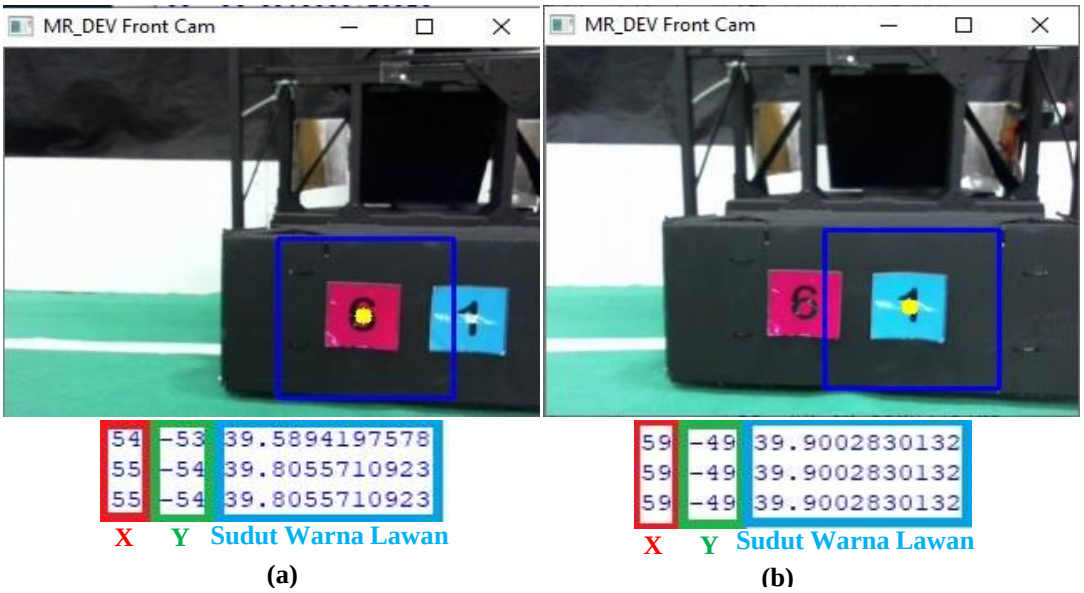
Berdasarkan hasil pengujian proses *thresholding*, proses ini telah berhasil mengembalikan citra yang pada tahap sebelumnya *blur* atau tersamarkan menjadi jelas kembali. Indikator tersebut membuktikan bahwa tahap ini telah dapat berjalan dengan baik. Selain itu, terlihat bahwa luas wilayah warna yang menjadi tanda warna lawan juga mengalami pembesaran, hal tersebut membuktikan bahwa tujuan dari proses filter 2D *Convolution* juga telah tercapai. Sehingga dengan demikian akan dapat meningkatkan jarak maksimum pendeteksian tanda warna lawan.



(a) **(b)**

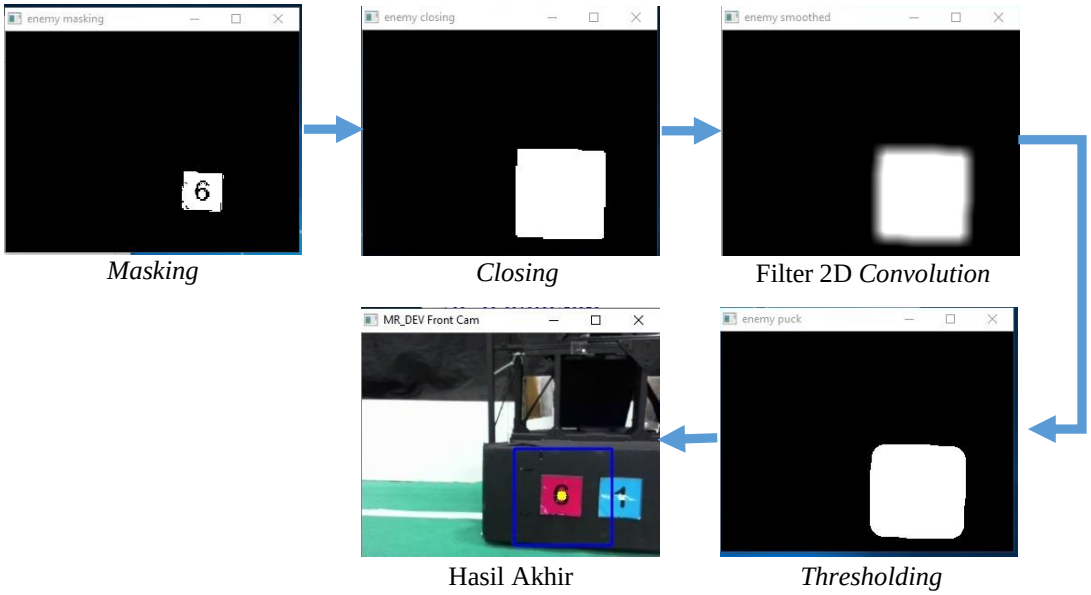
Gambar 4.46 Hasil Pengujian Proses *Thresholding* pada Deteksi Tanda Warna Lawan: **(a)**Magenta **(b)**Cyan

Pada tahap terakhir, proses yang dilakukan adalah menemukan kontur warna lawan, sehingga dengan demikian bisa mendapatkan posisi koordinat dari tanda warna lawan dengan menggunakan metode analisis *moment*.



Gambar 4.47 Hasil Akhir Deteksi Tanda Warna Lawan: **(a)**Magenta, **(b)**Cyan

Berikut adalah contoh tampilan secara keseluruhan hasil dari semua tahapan proses pendeteksian tanda warna lawan.



Gambar 4.48 Tahapan Proses Deteksi Tanda Warna Lawan

Berdasarkan hasil pengujian, proses tersebut telah dapat berjalan dengan baik, indikator yang menyatakan keberhasilan tersebut antara lain adalah,

1. Ketika tanda warna lawan berada tepat di tengah *frame*, nilai koordinat *x* telah berhasil mendekati nol (0).
2. Pada saat tanda warna lawan berada di sisi kanan *frame*, nilai koordinat *x* telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 159.
3. Sebaliknya, jika tanda warna lawan berada di sisi kiri *frame*, nilai koordinat *x* juga telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -159.
4. Ketika tanda warna lawan berada tepat di tengah *frame*, nilai koordinat *y* telah berhasil mendekati nol (0).
5. Pada saat tanda warna lawan berada di sisi atas *frame*, nilai koordinat *y* telah berhasil menunjukkan nilai positif, yaitu pada *range* 1 hingga 119.
6. Sebaliknya, saat tanda warna lawan berada di sisi bawah *frame*, nilai koordinat *y* telah berhasil menunjukkan nilai negatif, yaitu pada *range* -1 hingga -119.

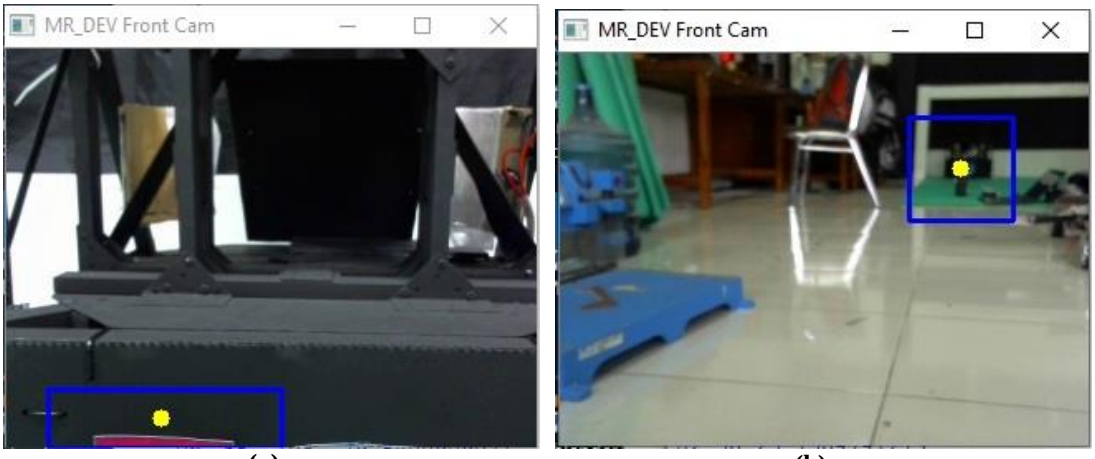
Pengujian selanjutnya adalah menguji jarak minimum dan maksimum tanda warna lawan dapat terdeteksi oleh sistem. Berikut adalah data hasil pengujian yang telah dilakukan.

Tabel 4.10 Hasil Pengujian Jarak Minimum dan Maksimum Deteksi Tanda Warna Lawan

No	Posisi Tanda Warna Lawan	Jarak	
		Minimum Deteksi	Maksimum Deteksi
1	0°	0,25 m	>7,2 m
2	10°	0,25 m	>7,2 m
3	20°	0,25 m	>7,2 m
4	30°	0,25 m	>7,2 m
5	40°	0,25 m	>7,2 m

Berdasarkan data pada tabel di atas, pengujian jarak minimum dan maksimum deteksi tanda warna lawan menunjukkan hasil yang positif. Hasil

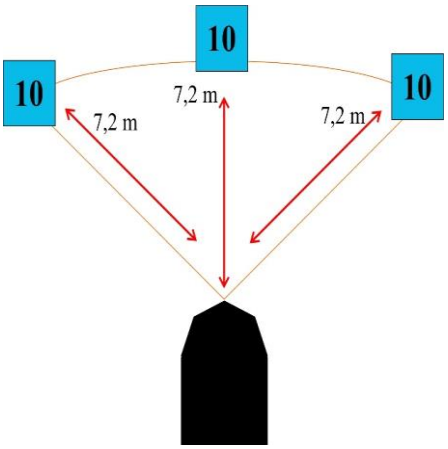
pengujian jarak maksimumnya bahkan menyamai hasil pengujian deteksi bola pada kamera depan. Jarak tersebut dapat tercapai dikarenakan nilai iterasi dari *dilate* pada proses *closing* terbilang cukup besar, sehingga luas wilayah tanda warna lawan menjadi meningkat pesat. Selain itu, faktor lain yang juga mendukung keberhasilan tersebut adalah hasil filter 2D *Convolution*, dimana menjadikan luas wilayah tanda warna lawan menjadi semakin besar.



(a) **(b)**

Gambar 4.49 Contoh Pengujian Jarak **(a)**Minimum dan **(b)**Maksimum Deteksi Tanda Warna Lawan

Gambar 4.50 merupakan ilustrasi jangkauan deteksi tanda warna lawan pada kamera depan.



Gambar 4.50 Ilustrasi Jangkauan Jarak Terjauh Deteksi Tanda Warna Lawan

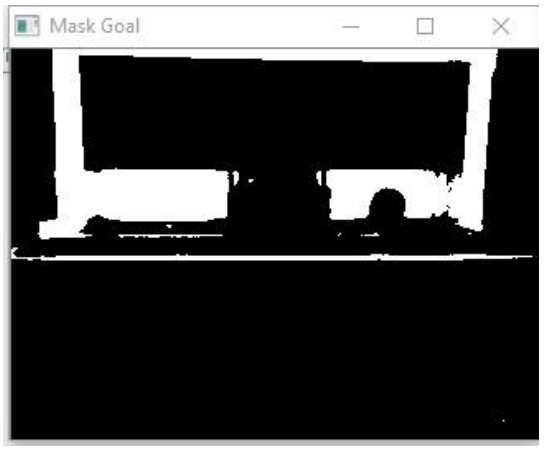
4.9 Pengujian Deteksi Gawang Lawan

Pengujian deteksi Gawang Lawan juga diawali dengan proses penyetelan nilai parameter, tetapi bedanya adalah tidak melakukan *tuning* nilai HSV, melainkan nilai *grayscale.* Pada deteksi gawang lawan, dipilih menggunakan ruang warna *grayscale* dikarenakan dalam ruang warna tersebut, *range* warnanya hanya 0 (hitam) hingga 255(putih), sehingga dengan demikian akan lebih mudah dalam mencari *range masking* warna putih pada gawang lawan daripada dengan menggunakan HSV. Berikut adalah hasil dari proses *tuning* warna *grayscale.*

Tabel 4.11 Hasil Penyetelan Nilai *Grayscale* Warna Gawang Lawan

No	Parameter	Nilai Parameter *Grayscale* Gawang
1	*White High*	255
2	*White Low*	165

Terlihat dari **Gambar 4.51**, proses *masking* warna putih dapat berjalan dengan baik, hal tersebut menandakan hasil *tunning* parameter citra *grayscale* pada warna gawang telah tepat. Terlihat pada **Gambar 4.51**, terdapat warna hitam yang berukuran cukup besar diantara warna putih, dimana warna hitam tersebut adalah robot penjaga gawang lawan dan juga sebuah bola.



Gambar 4.51 Hasil Proses *Masking* Warna Gawang Lawan

Tahap selanjutnya adalah proses *opening*, dimana pada tahap ini nilai kernel yang digunakan sama seperti halnya pada proses deteksi objek-objek lainnya. Perbedaan proses *opening* dalam deteksi gawang lawan adalah dimana jumlah iterasi antara *erode* dan *dilate* yang digunakan berbeda nilainya. Nilai iterasi yang

erode yang digunakan adalah 5 kali, sedangkan *dilate* sebanyak 10 kali. Nilai iterasi proses erosi tersebut mengakibatkan sebagian wilayah warna gawang lawan menjadi tereliminasi, setelah itu proses dilasi memperluas wilayah warna gawang yang berhasil bertahan dari proses eliminasi sebelumnya. Terlihat dari hasil pengujian bahwa bagian tiang dan mistar gawang tereliminasi pada saat proses erosi. Selanjutnya bagian yang bertahan dari proses erosi, yaitu bagian belakang gawang, mengalami perluasan wilayah setelah melalui proses dilasi. Hal tersebut sengaja dilakukan dengan tujuan untuk kemudian dapat memperoleh titik tembak terbaik pada gawang lawan, dimana acuan yang digunakan sebagai titik tembak adalah berdasarkan luas wilayah warna bagian belakang gawang.

Gambar 4.52 Hasil Proses *Opening* Warna Gawang Lawan

Tahap berikutnya adalah *closing*, dimana berfungsi untuk menutupi warna Seperti pada proses deteksi objek lainnya. Pada tahap ini nilai kernel yang digunakan adalah matriks berukuran 30 kali 30. Sedangkan nilai iterasi antara *dilate* dan *erode* yang digunakan sama, sehingga *output*nya tidak akan merubah luas wilayah warna gawang atau dengan kata lain sama seperti tahap *opening.* Terlihat dari hasil keluarannya, garis hitam yang terdapat pada tahap sebelumnya telah tidak lagi terlihat atau tertutupi. Hal tersebut menunjukkan bahwa tujuan dari tahap *closing* telah tercapai.

Gambar 4.53 Hasil Proses *Closing* Warna Gawang Lawan

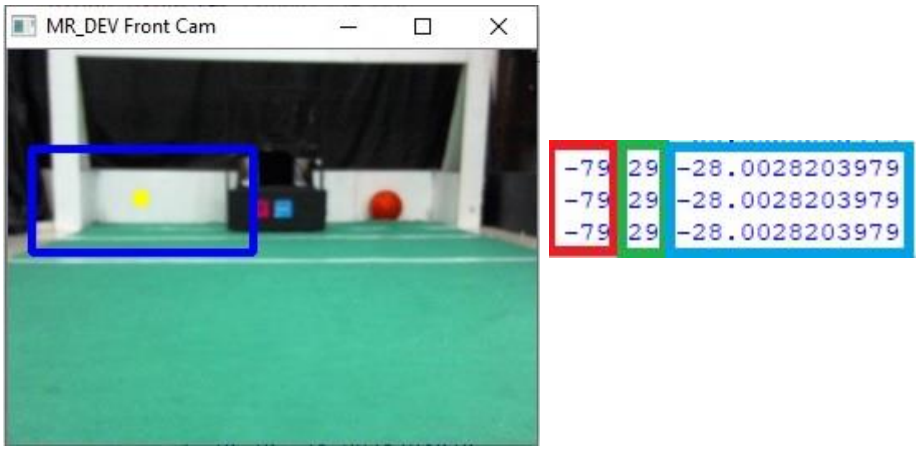
Pada tahap filter 2D *Convolution*, terlihat bahwa citra keluaran filter berubah menjadi tersamarkan atau *blur*. Hal tersebut menunjukkan bahwa proses ini telah dapat berjalan sebagai mana mestinya.

Gambar 4.54 Hasil Proses 2D *Convolution* Warna Gawang Lawan

Seperti pada proses deteksi objek-objek lain dalam penelitian ini, tahap *thresholding* bertujuan untuk mengembalikan citra *blur* kembali menjadi jelas. Terlihat dari hasil proses *thresholding*, citra telah berhasil kembali menjadi jelas kembali. Selain itu juga terlihat bahwa bahwa luas wilayah warna gawang menjadi lebih luas dari proses sebelumnya, hal tersebut juga menunjukkan bahwa tujuan dari diadakannya proses filter pada deteksi gawang lawan ini telah berhasil, yaitu dimana untuk mempertebal luas wilayah warna gawang dan mengurangi tingkat *noise* disekitar warna gawang.

Gambar 4.55 Hasil Proses *Thresholding* Warna Gawang Lawan

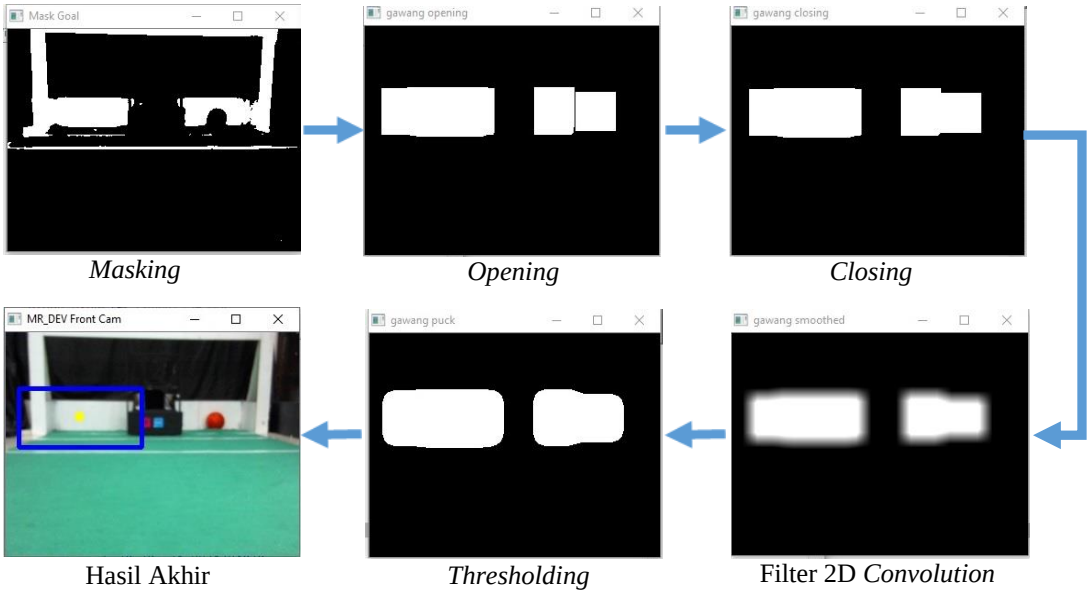
Terlihat dari hasil proses sebelumnya, bahwa terdapat dua buah wilayah warna gawang yang terpisahkan oleh sekat berwarna hitam yang mana merupakan robot penjaga gawang lawan. Kedua wilayah warna gawang tersebut selanjutya dicari konturnya masing-masing. Setelah didapatkan kedua kontur tersebut, algoritma yang digunakan untuk menentukan titik tembak dilakukan dengan cara memilih kontur terbesar di antara kedua kontur tersebut. Dipilih kontur terbesar karena itu artinya daerah tersebut lebih jauh posisinya terhadap robot penjaga gawan lawan daripada yang memiliki kontur lebih kecil. Jadi dengan demikian akan lebih besar peluangnya untuk dapat mencetak gol ke gawang lawan. Langkah selanjutnya setelah menentukan kontur terbesar adalah menentukan titik tengah atau *center of gravity* dari kontur tersebut. Titik tengah tersebutlah yang kemudian akan menjadi koordinat titik tembak pada gawang lawan.



Gambar 4.56 Hasil Akhir Deteksi Titik Tembak pada Gawang Lawan

Selain data koordinat titik tembak gawang lawan, hasil keluaran dari proses deteksi gawang lainnya adalah berupa sudut titik tembaknya. Sehingga dengan demikian, ketika robot sudah mendapatkan bola dan berada diposisi lebih dari garis tengah lapangan, robot tidak perlu bergeser untuk mencapai titik tembak, melainkan cukup dengan memutar saja. Algoritma tersebut akan membuat respon robot penjaga gawang lawan menjadi lebih terlambat dibandingkan dengan harus bergeser terlebih dahulu untuk mencapai titik tembak. Algoritma pembacaan sudut titik tembak sama persis seperti yang digunakan dalam pembacaan sudut bola kamera depan.

Berikut adalah tampilan secara keseluruhan hasil dari seluruh tahapan proses pendeteksian gawang lawan.



Gambar 4.57 Tahapan Proses Deteksi Titik Tembak pada Gawang Lawan

4.10 Pengujian Nilai FPS pada Kamera Depan dan Omnivision

Pada pengujian nilai FPS kamera depan, sistem *image processing* Mr Dev mendapatkan rata-rata nilai FPS (*Frame Per Second*) sebesar 25,324. Pada tingkatan rata-rata nilai FPS tersebut, kualitas *streaming* video hasil pengolahan citra masih tergolong halus atau tidak patah-patah. Jadi dengan demikian pengolahan citra pada kamera depan dapat berjalan dengan cukup cepat.

Tabel 4.12 Hasil Pengujian Nilai FPS pada Kamera Depan

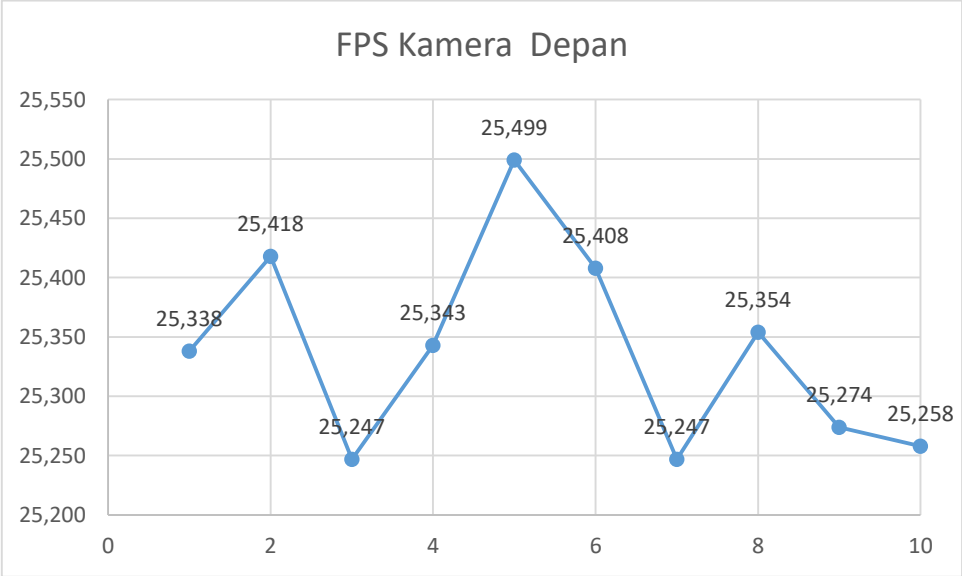
Pengujian ke-	Jumlah Frame	Waktu yang Dibutuhkan (detik)	Hasil FPS
1	120	4,763 s	25,338
2		4,721 s	25,418
3		4,753 s	25,247
4		4,735 s	25,343
5		4,706 s	25,499
6		4,723 s	25,408
7		4,753 s	25,247
8		4,733 s	25,354
9		4,748 s	25,274
10		4,751 s	25,258
Rata - Rata		4,7386 s	25,324

Gambar 4.58 merupakan contoh hasil pengujian FPS pada kamera depan.

```
Capturing 120 frames
Time taken : 4.70599985123 seconds
Estimated frames per second : 25.4993633221
```

Gambar 4.58 Contoh Hasil Pengujian FPS pada Kamera Depan

Grafik 4.3 menunjukkan hasil pengujian nilai FPS pada kamera depan dalam 10 kali percobaan.



Grafik 4.3 Hasil Pengujian FPS pada Kamera Depan dalam 10 Kali Percobaan

Sedangkan pada pengujian nilai FPS kamera omnivision mendapatkan rata-rata nilai FPS sebesar 20,174. Pada rentang nilai tersebut, kualitas streaming video hasil pengolahan citra terlihat cukup halus atau tidak patah-patah. Sehingga dengan demikian, pengolahan citra pada kamera omnivision dapat berjalan dengan cukup cepat.

Tabel 4.13 Hasil Pengujian Nilai FPS pada Kamera Omnivision

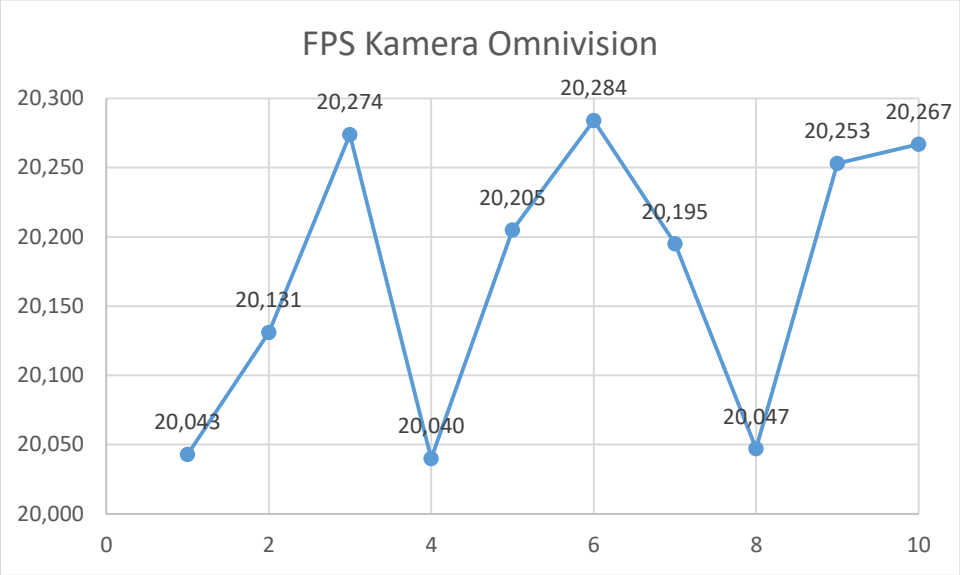
Pengujian ke-	Jumlah Frame	Waktu yang Dibutuhkan (detik)	Hasil FPS
1	120	5,987 s	20,043
2		5,961 s	20,131
3		5,919 s	20,274
4		5,988 s	20,040
5		5,939 s	20,205
6		5,916 s	20,284
7		5,942 s	20,195
8		5,986 s	20,047
9		5,925 s	20,253
10		5,921 s	20,267
Rata - Rata		5,9484 s	20,174

Gambar 4.59 merupakan contoh hasil pengujian FPS pada kamera omnivision.

```
Capturing 120 frames
Time taken : 5.91599988937 seconds
Estimated frames per second : 20.2839760385
```

Gambar 4.59 Contoh Hasil Pengujian FPS pada Kamera Omnivision

Grafik 4.4 menunjukkan hasil pengujian nilai FPS pada kamera omnivision yang mana dilakukan dalam 10 kali pengambilan data.



Grafik 4.4 Hasil Pengujian FPS pada Kamera Omnivision dalam 10 Kali Percobaan

4.11 Pengujian Pengiriman Data ke *Microcontroller*

Data-Data keluaran dari sistem *image processing* yang dikirimkan ke *microcontroller* melalui komunikasi UART antara lain adalah data koordinat dan sudut bola pada kamera depan, koordinat dan sudut bola pada kamera omnivision, koordinat dan sudut titik tembak gawang lawan, serta yang terakhir adalah koordinat dan sudut lawan terhadap robot Mr Dev. Data-data tersebut sangat penting untuk dikirimkan karena akan sangat membantu sistem control untuk menentukan respon robot terhadap segala situasi yang mungkin akan terjadi. Sebagai contoh, data koordinat dan sudut bola akan sangat membantu robot untuk mendapatkan bola. Data koordinat dan sudut titik tembak pada gawang lawan akan sangat membantu robot untuk menentukan posisi robot akan menembakkan bola, dan lain sebagainya. Oleh karena itulah pengiriman data merupakan salah satu hal vital yang wajib untuk diperhatikan.

Berdasarkan hasil pengujian pengiriman data, terlihat bahwa tidak ada data yang saling bertumbukan atau tercampur antara satu sama lain. Selain itu, data yang diterima pun juga tidak berubah urutannya, atau dengan kata lain sama seperti pada saat dikirimkan, yaitu dengan urutan dimulai dari koordinat *x*, *y* dan sudut bola pada kamera depan, kemudian koordinat *x*, *y* dan sudut bola pada kamera omnivision,

selanjutnya koordinat *x*, *y* dan sudut titik tembak ke gawang lawan, serta terakhir adalah koordinat *x*, *y*, dan sudut tanda warna lawan, sehingga dengan demikian total terdapat 12 data yang dikirimkan ke *microcontroller* secara bersamaan. Jadi dengan demikian tidak terjadi *corrupt* data selama proses pengiriman.

```
28.00 -77.00 33.00 -5.00 84.00 -3.00 -59.00 41.00 -20.00 92.00 8.00 35.00
28.00 -77.00 33.00 -5.00 85.00 -3.00 -59.00 41.00 -20.00 92.00 8.00 35.00
28.00 -77.00 33.00 -5.00 84.00 -3.00 -59.00 41.00 -20.00 92.00 8.00 35.00
28.00 -77.00 33.00 -5.00 84.00 -3.00 -59.00 41.00 -20.00 92.00 8.00 35.00
```

Gambar 4.60 Tampilan Data saat Berhasil Diterima Kembali pada Sistem