BAB V

IMPLEMENTASI DAN PENGUJIAN SISTEM

5.1 HASIL IMPLEMENTASI PROGRAM

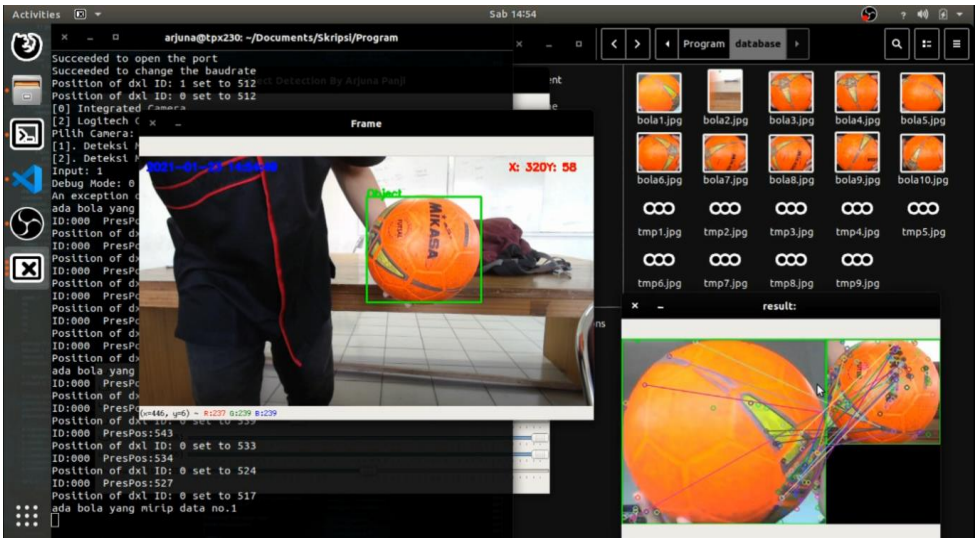
Sistem pendeteksian objek bola menggunakan algoritma *SIFT* dan *Kalman Filter* ini terdiri dari perancangan *software* dan *hardware.* Perancangan *software* ditulis menggunakan bahasa pemrograman *python* yang dijalankan pada sistem operasi *Linux*.

Tahap – tahap implementasi adalah sebagai berikut :

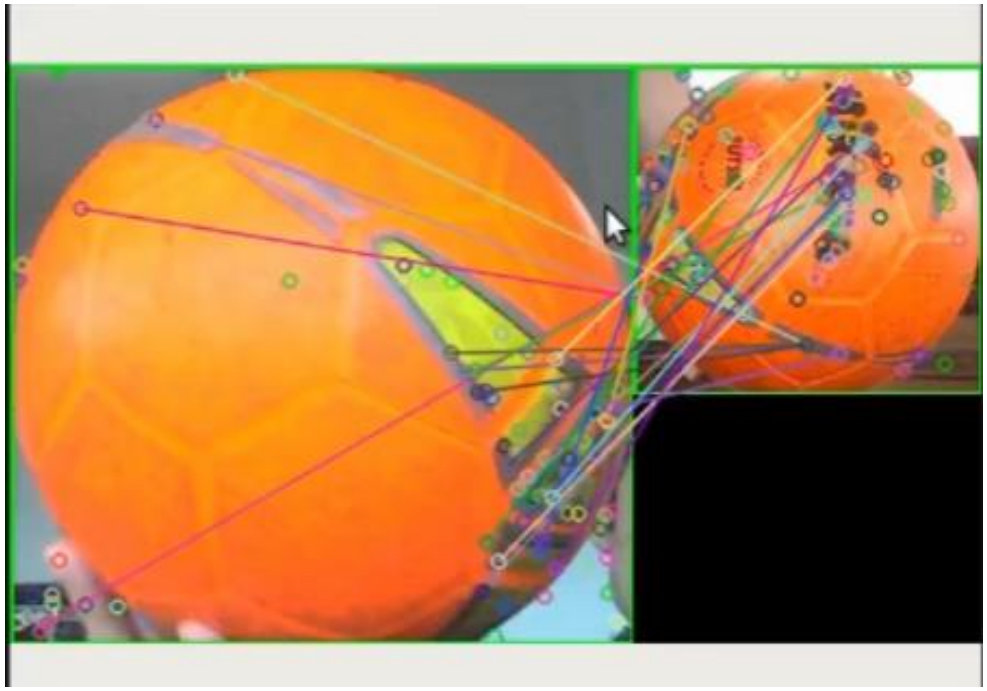
1. Menyiapkan *virtual environtment* beserta *libraries* yang diperlukan.
2. Menulis kode *program* pendeteksian objek dalam *visual studio code*.
3. Memasukkan algoritma *SIFT* dan algoritma *Kalman Filter* kedalam *program* pendeteksian objek.
4. Menyambungkan *Servo Dynamixel AX-12A* ke Laptop menggunakan *USB2Dynamixel*.
5. Menguji *program* pendeteksian objek yang telah dibuat.

5.1.1 Implementasi Algoritma *SIFT*

Algoritma *SIFT* bekerja dengan cara menandai *feature* lokal dalam matriks citra yang disebut *keypoint* dan kemudian membandingkan apakah terdapat kemiripan *keypoint* dengan sampel yang telah disiapkan. Berikut hasil implementasi algoritma *SIFT* dapat dilihat pada gambar 5.1 dan 5.2.



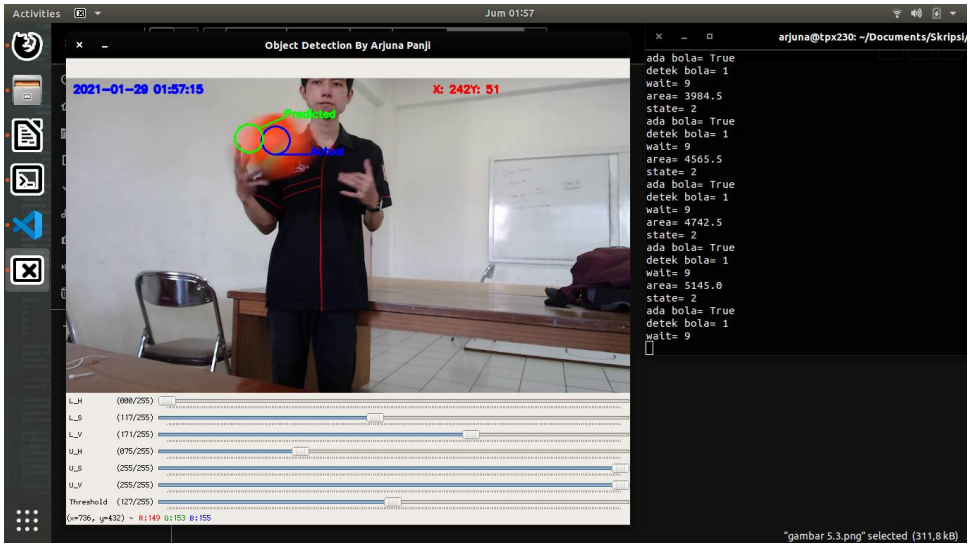
Gambar 5.1 Pendeteksian Menggunakan Algoritma SIFT



Gambar 5.2 Membandingkan Keypoint Dari Kedua Citra

5.1.2 Implementasi Algoritma Kalman Filter

Algoritma *kalman filter* berfungsi untuk memprediksi *state* objek berdasarkan *state* sebelumnya. Untuk hasil implementasi algoritma *kalman filter* dapat dilihat pada gambar 5.3 berikut:



Gambar 5.3 Pendeteksian Menggunakan Algoritma Kalman Filter

5.1.3 Tampilan Program Pendeteksian

Tampilan program dibuat minimalis karena program ini ditujukan untuk robot otomatisasi sehingga tidak banyak *user action* yang dilakukan setelah program dijalankan. Berikut tampilan program dapat dilihat pada gambar 5.4.

Gambar 5.4 User Interface

5.2 PENGUJIAN PERANGKAT LUNAK

5.2.1 Pengujian Komunikasi Serial

Pada tahap ini, dilakukan pengujian pada modul komunikasi serial untuk mengetahui apakah alat terhubung ke sistem berjalan dengan baik atau tidak. Hasil pengujian pada modul komunikasi serial dapat dilihat pada Tabel 5.1.

Tabel 5.1 Pengujian Modul Komunikasi Serial

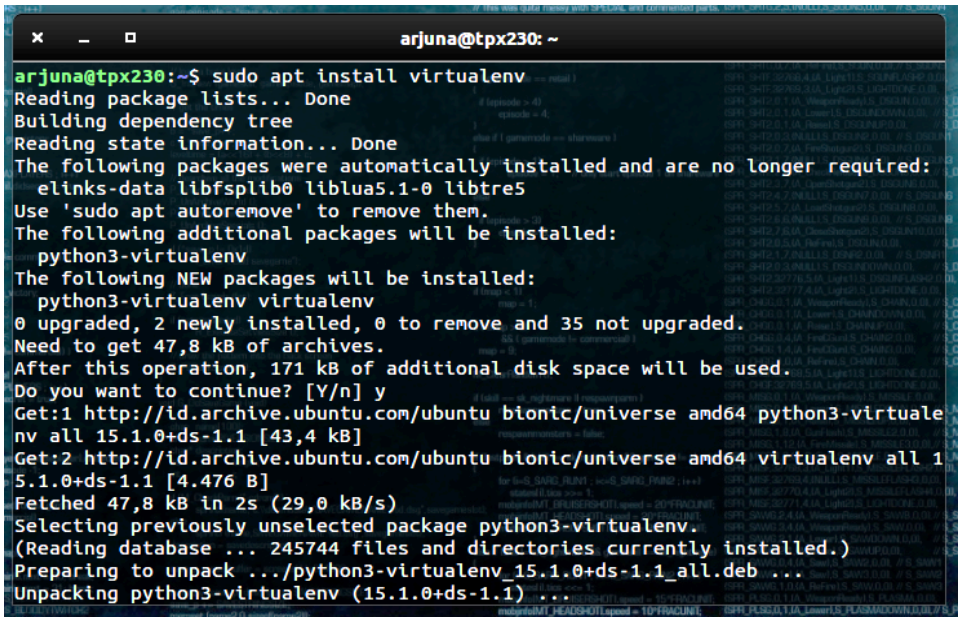
Modul yang diuji	Prosedur Pengujian	Masukan	Keluaran yang di harapkan	Hasil yang didapat	Kesimpulan
Komunikasi serial	Menghubungkan port serial yang tersambung pada Servo Dynamixel AX-12A dengan laptop/pc menggunakan USB2Dynamixel.	Mengecek apakah port berhasil tersambung.	Port serial yang terhubung dapat terdeteksi.	Port serial berhasil terhubung dan dapat digunakan.	Baik
Servo AX-12A	Memposisikan servo ke posisi center.	Mengatur goal position servo ke 512.	Servo kembali ke posisi center.	Perintah berhasil terkirim dan servo kembali ke posisi center.	Baik
Servo AX-12A	Menggerakkan servo dari kiri ke kanan.	Memasukkan goal position servo.	Servo bergerak dari kiri ke kanan.	Servo berhasil bergerak dari kiri ke kanan.	Baik
Servo AX-12A	Menggerakkan servo dari atas ke bawah.	Memasukkan goal position servo.	Servo bergerak dari atas ke bawah	Servo berhasil bergerak dari atas ke bawah	Baik

5.3 PENGUJIAN SISTEM

Pengujian sistem bertujuan untuk memastikan apakah semua fungsi sistem bekerja dengan baik dan mencari kesalahan yang mungkin terjadi. Dalam pengujian sistem meliputi pengujian perangkat lunak dan pengujian perangkat keras.

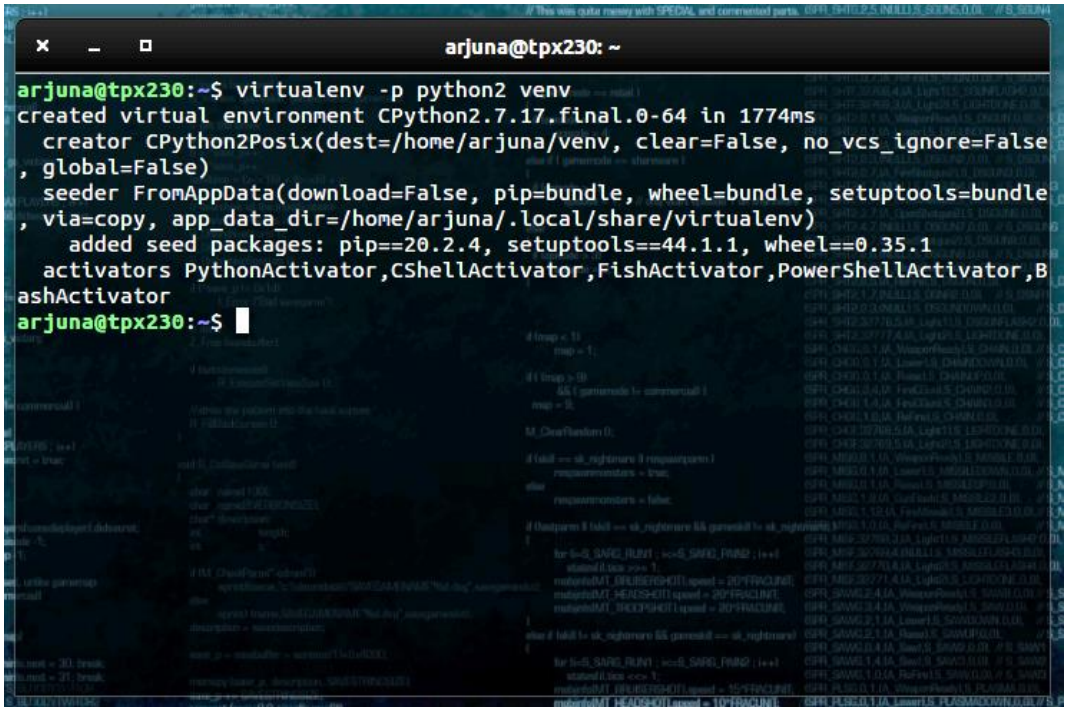
5.3.1 *Virtual Environtment*

Sebelum membuat sebuah *project*, hal pertama yang harus dilakukan yaitu membuat *virtual environtment*, hal ini dilakukan agar pustaka yang digunakan tidak tercampur dengan pustaka pada *project* lain. Caranya pertama instal *virtualenv* melalui terminal ketik perintah ***sudo apt install virtualenv*** seperti pada gambar 5.5.



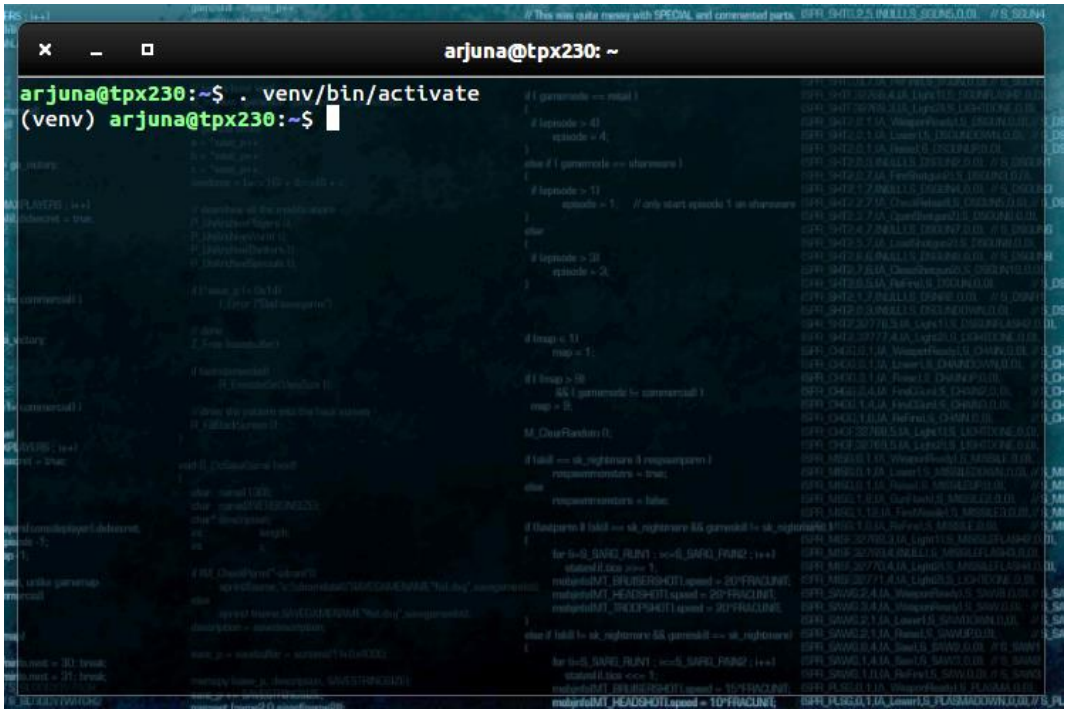
Gambar 5.5 Instalasi Virtual Environtment Python

Setelah berhasil menginstal *virtualenv*, selanjutnya membuat *virtual environtment* dengan perintah **virtualenv -p <versi python> <nama virtual>** seperti pada gambar 5.6.



Gambar 5.6 Membuat Virtual Environtment Kemudian untuk masuk kedalam virtual environtment dapat dilakukan dengan perintah **. <nama virtual>/bin/activate** atau **source <nama virtual>/bin/activate** seperti pada gambar 5.7.

Kemudian untuk masuk kedalam virtual environtment dapat dilakukan dengan perintah **. <nama virtual>/bin/activate** atau **source <nama virtual>/bin/activate** seperti pada gambar 5.7.



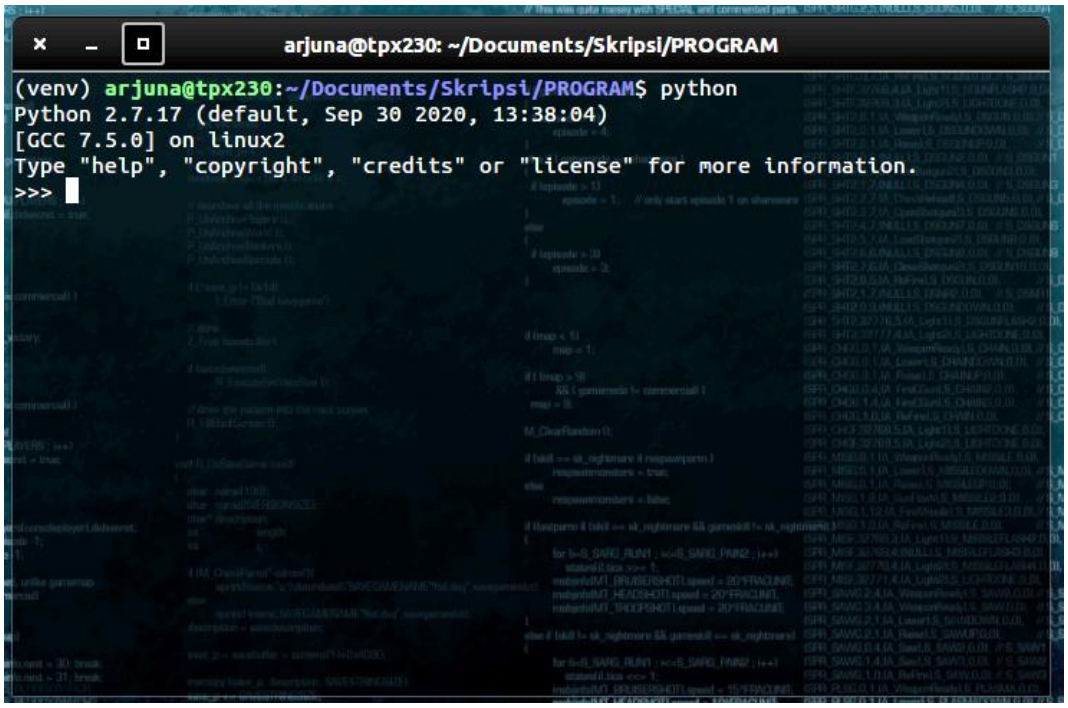
Gambar 5.7 Mengaktifkan Virtual Environtment Setelah masuk kedalam *virtual environtment*, langkah selanjutnya adalah meng-*install* semua *library* yang dibutuhkan dengan perintah **pip install <nama library>.**

Setelah masuk kedalam *virtual environtment*, langkah selanjutnya adalah meng-*install* semua *library* yang dibutuhkan dengan perintah **pip install <nama library>.**

5.3.2 Python

Dalam sistem pendeteksian objek ini, peneliti menggunakan bahasa pemrograman *python*.

Untuk pengujian, dapat dimulai dari menjalankan *python* melalui terminal dengan mengetik *python* kemudian *enter* seperti pada gambar 5.8.

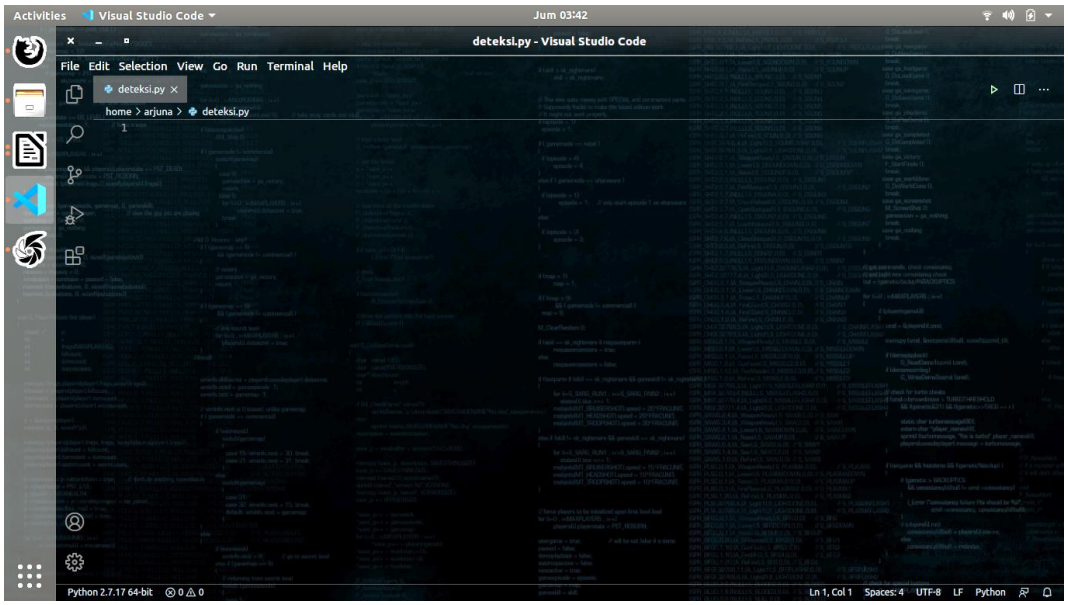


Gambar 5.8 Menjalankan Python* pada *Terminal

Bila tampilannya sudah seperti pada gambar diatas, tandanya *python* telah berhasil di instal. Selanjutnya menginstall semua library yang dibutuhkan dalam penelitian ini yaitu, OpenCV, Numpy, Matplotlib.

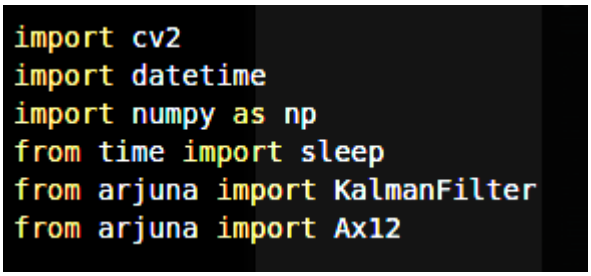
5.3.3 Visual Studio Code

Setelah semua library telah di instal, selanjutnya dimulai penulisan kode sistem pendeteksian objek menggunakan text editor Visual Studio Code, buka VSCode dan buat file baru kemudian simpan dengan extension .py seperti pada gambar 5.9.



Gambar 5.9 Tampilan Visual Studio Code

Kemudian *import library* yang diperlukan seperti yang dapat dilihat pada gambar 5.10 berikut.



Gambar 5.10 Import Library Python

5.3.4 Menjalankan Program

Jika penulisan kode telah selesai, langkah selanjutnya yaitu menjalankan program, hal ini bisa dilakukan dengan dua cara yaitu dengan menekan F5 jika didalam VSCode, atau bisa dengan mengetik **python <nama file>.py** pada terminal untuk menjalankan program.

5.4 PENGUJIAN ALAT

Adapun rancangan alat yang digunakan dalam simulasi ini adalah sebagai berikut :

Gambar 5.11 Bentuk Fisik Webcam dan Servo

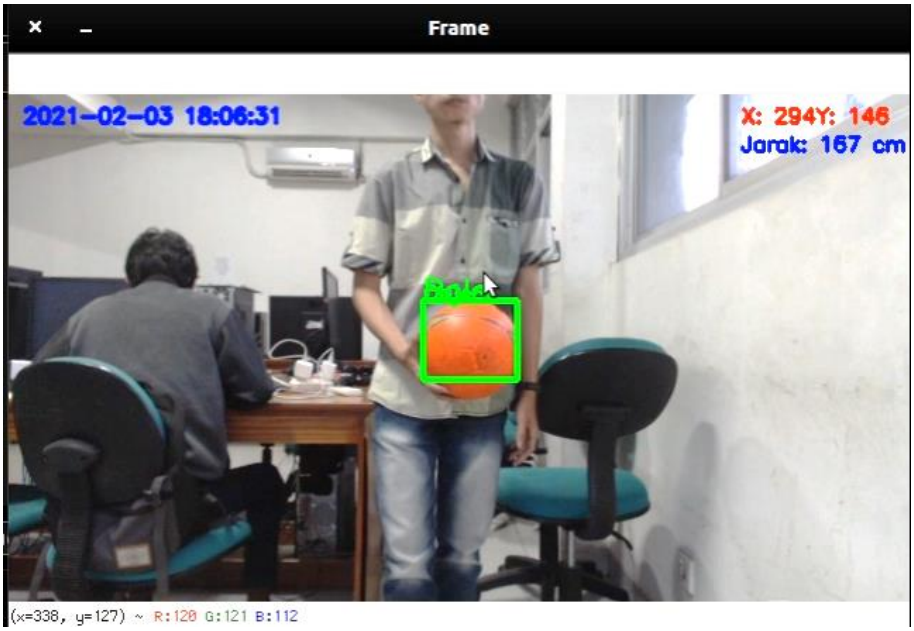
5.4.1 Pengujian Servo Dynamixel AX-12A

Pengujian dilakukan dengan cara mengubah nilai goal position pada servo dan melihat reaksi dari servo tersebut apakah ada kendala atau tidak agar kamera tidak kesulitan saat mengikuti pergerakan dari objek.

Tabel 5.2 Pengujian Servo Dynamixel AX-12A

ID Servo	Prosedur Pengujian	Masukan	Keluaran yang di harapkan	Hasil yang didapat	Kesimpulan
00	Mengatur Goal Position -90°	Goal Position di atur ke -90°	Servo bergerak kearah -90°	Servo berhasil bergerak kearah -90°	Baik
00	Mengatur Goal Position +90°	Goal Position di atur ke +90°	Servo bergerak kearah +90°Servo berhasil bergerak kearah -90°	Servo berhasil bergerak kearah +90°	Baik
01	Mengatur Goal Position -90°	Goal Position di atur ke -90°	Servo bergerak kearah -90°	Servo berhasil bergerak kearah -90°	Baik
01	Mengatur Goal Position +90°	Goal Position di atur ke +90°	Servo bergerak kearah +90°	Servo berhasil bergerak kearah +90°	Baik

5.4.2 Pengujian Deteksi Menggunakan Algoritma SIFT



Gambar 5.12 Pengujian Jarak Deteksi Dengan SIFT

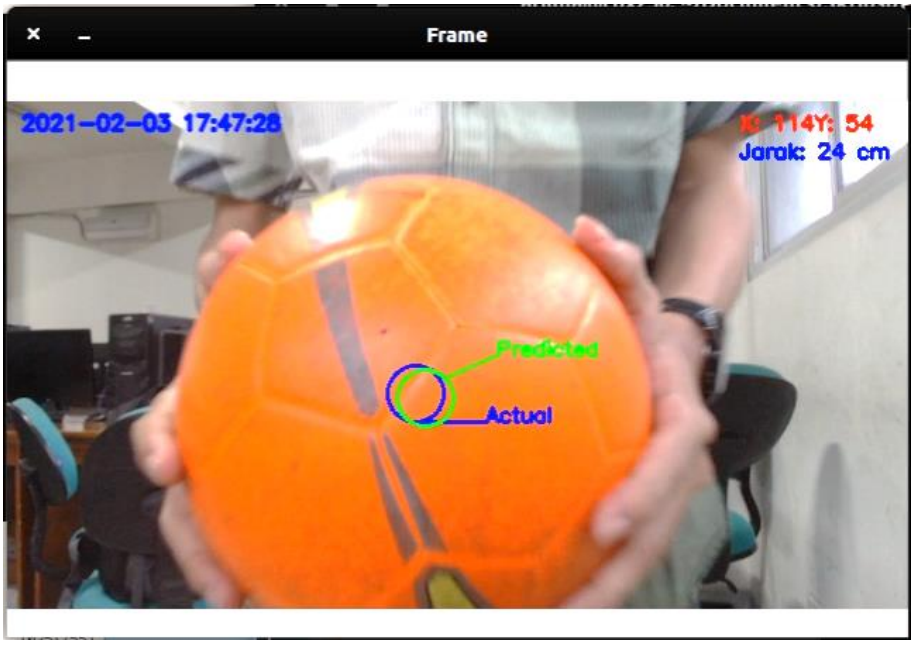
Pengujian ini dilakukan dengan tujuan untuk melihat sejauh mana kamera webcam dapat mendeteksi objek dengan baik saat menggunakan algoritma *SIFT*. Hasil pengujian dapat dilihat pada tabel 5.3.

Tabel 5.3 Pengujian Jarak Deteksi dengan Algoritma SIFT

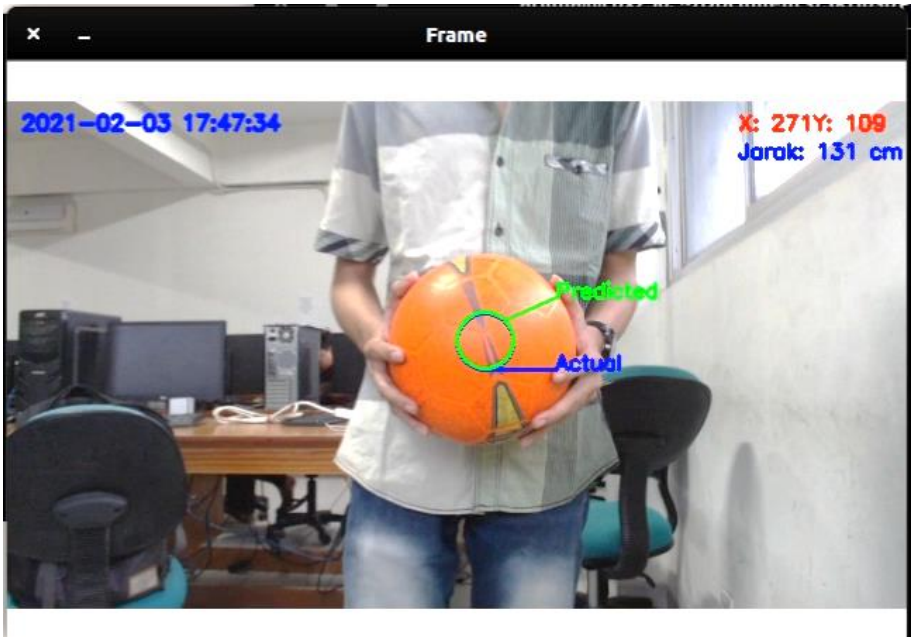
Percobaan	Jarak	Koordinat		Kesimpulan
		x	y	
1	30 cm	120	42	Berhasil
2	50 cm	321	44	Berhasil
3	70 cm	294	84	Berhasil
4	90 cm	275	65	Berhasil
5	110 cm	282	132	Berhasil
6	130 cm	190	87	Berhasil

7	150 cm	287	68	Berhasil
8	170 cm	291	121	Berhasil
9	190 cm	342	144	Berhasil
10	210 cm	-	-	Tidak Berhasil

5.4.3 Pengujian Deteksi Menggunakan Algoritma Kalman Filter



Gambar 5.13 Pengujian Jarak Deteksi Dengan Algoritma Kalman Filter



Gambar 5.14 Pengujian Jarak Deteksi Dengan Algoritma Kalman Filter

Gambar 5.15 Pengujian Jarak Deteksi Dengan Algoritma Kalman Filter

Pengujian ini dilakukan untuk melihat hasil dari pengaplikasian algoritma *Kalman Filter* dalam sistem pendeteksian bola. Hasil pengujian dapat dilihat pada tabel 5.4 dan 5.5.

Tabel 5.4 Pengujian Jarak Deteksi dengan Kalman Filter

Percobaan	Jarak	Koordinat		Kesimpulan
		x	y	
1	30 cm	154	44	Berhasil
2	50 cm	186	33	Berhasil
3	70 cm	222	40	Berhasil
4	90 cm	255	56	Berhasil
5	110 cm	294	73	Berhasil
6	130 cm	174	93	Berhasil
7	150 cm	390	84	Berhasil
8	170 cm	318	145	Berhasil
9	190 cm	217	112	Berhasil
10	210 cm	-	-	Tidak Berhasil

Tabel 5.5 Pengujian Pembacaan Koordinat dengan Kalman Filter

Percobaan Ke-	Nilai Koordinat (Piksel)					
	Aktual		Prediksi		Error Prediksi	
	x	y	x	y	x	y
1	132	192	0	0	132	192
2	130	186	65	93	65	93
3	130	180	130	180	0	0
4	133	174	152	197	-19	-23

5	140	167	156	182	-16	-15
6	164	150	180	151	-16	-1
7	183	140	201	133	-18	7
8	207	129	227	119	-20	10
9	238	114	261	102	-23	12
10	267	104	294	91	-27	13
11	322	96	356	85	-34	11
12	346	91	383	82	-37	9
13	365	94	395	89	-30	5
14	383	96	407	94	-24	2
15	398	104	417	106	-19	-2
16	424	117	443	123	-19	-6
17	432	123	449	131	-17	-8
18	439	128	452	135	-13	-7
19	442	131	450	136	-8	-5
20	443	133	448	137	-5	-4
21	440	129	441	130	-1	-1
22	436	126	434	124	2	2
23	430	121	426	118	4	3
24	423	114	417	109	6	5
25	413	106	406	99	7	7
26	382	90	368	80	14	10

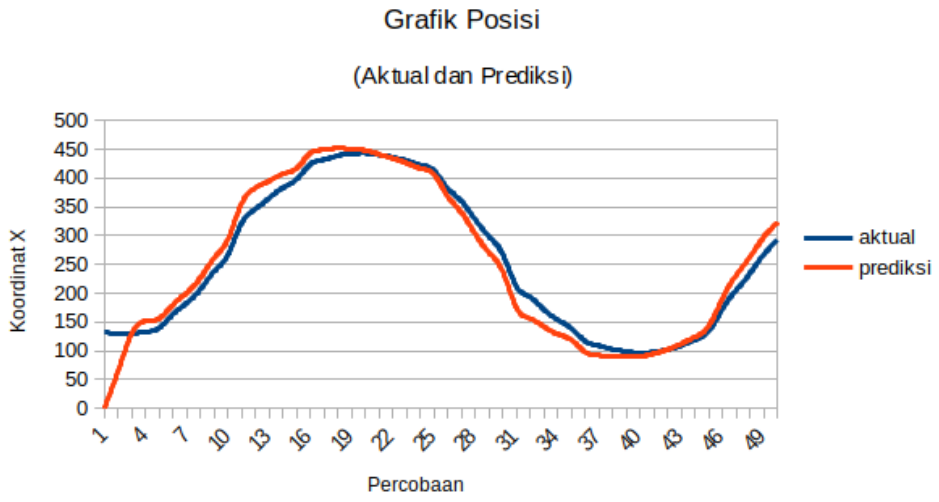
27	360	79	340	67	20	12
28	328	71	303	60	25	11
29	299	67	271	58	28	9
30	268	66	238	60	30	6
31	212	74	175	73	37	1
32	193	79	156	81	37	-2
33	172	86	142	90	30	-4
34	154	94	129	100	25	-6
35	139	103	119	110	20	-7
36	117	117	98	126	19	-9
37	109	123	92	132	17	-9
38	103	126	90	133	13	-7
39	99	129	90	134	9	-5
40	96	129	90	132	6	-3
41	98	126	95	126	3	0
42	102	122	102	120	0	2
43	109	116	112	112	-3	4
44	119	108	124	102	-5	6
45	134	100	143	93	-9	7
46	173	79	191	68	-18	11
47	205	67	231	53	-26	14
48	234	61	263	49	-29	12

49	267	56	297	47	-30	9
50	293	57	323	51	-30	6

Dari nilai koordinat aktual dan prediksi yang dihasilkan oleh *Kalman filter*, terdapat *error* pada masing-masing percobaan. Perhitungan nilai *error* bertujuan untuk melihat perbedaan antara nilai koordinat aktual dengan nilai koordinat prediksi. Untuk menghitung nilai *error* digunakan rumus sebagai berikut:

- *Error* prediksi data ke-i = nilai koordinat aktual data ke-i – nilai koordinat prediksi data ke-i
- Rata-rata *error* prediksi = $\frac{\Sigma_{i=1}^{n} error_{prediksi}}{n}$
- Rata-rata *error* prediksi pada sumbu x = $\frac{\Sigma_{i=1}^{n} error_{prediksi}}{n}$ = 1.06
- Rata-rata *error* prediksi pada sumbu y = = 7.34

Berdasarkan hasil pengujian pendeteksian objek menggunakan *kalman filter* maka diperoleh hasil seperti dapat dilihat pada gambar 5.16.



Gambar 5.16 Grafik Posisi (Aktual dan Prediksi) *Kalman Filter*

5.5 ANALISIS SISTEM SECARA KESELURUHAN

Untuk mendeteksi apabila terjadi kesalahan setelah uji coba, maka perlu dilakukan analisa rangkaian secara keseluruhan. Dari seluruh proses yang telah dilakukan, baik pengujian perangkat keras maupun perangkat lunak, dapat dikatakan bahwa alat ini berfungsi sebagai yang penulis inginkan.

Pengujian ini dilakukan untuk mencoba dan membuktikan apakah algoritma ini dapat di gunakan untuk sistem pendeteksian robot sepak bola beroda dan mencari tahu apakah dengan menggunakan algoritma ini robot sepak bola beroda dapat mendeteksi objek dengan lebih akurat. Pengujian ini dilakukan dengan cara sebagai berikut :

1. Menghubungkan Servo Dynamixel AX-12A ke PowerHub
2. Memasang USB2Dynamixel ke PC/Laptop

3. Menjalankan program *python* yang telah penulis buat.
4. Input “1” untuk deteksi menggunakan algoritma *SIFT*
5. Input “2” untuk deteksi menggunakan algoritma *Kalman Filter*
6. Lalu selanjutnya koordinat (X,Y) dari objek akan dikirimkan melalui serial