DAFTAR PUSTAKA

Afrisal, Hadha, Faris, Muhammad, Utomo, Guntur P. Dkk. 2013. *Portable Smart Sorting And Grading Machine Machine For Fruits Using Computer Vision*. International Conference on Computer, Control, Informatics and Its Applications.

Arakeria, Megha.P. , Lakshmana. 2016. *Computer Vision Based Fruit Grading System For Quality Evaluation Of Tomato In Agriculture Industry*. 7th International Conference on Communication, Computing and Virtualization 2016.

Argo, B.D., Yogantoro, N. 2007. *Perancangan Sistem Kendali Konveyor Mikrokontroller AT89C51 untuk Sortasi Jeruk Manis (Citrus sinesis L.) berbasis Citra.* Jurnal Teknologi Pertanian, Vol. 8. No. 1 (April 2007) 26 – 34.

Aziz, A. H. Abdul, Ismail, A. H., dkk. 2014. *Design Of A Capacitive Sensor For Oil Palm Fruit Maturity Grading*. 2014 2nd International Conference on Electronic Design (ICED).

Baker, Ian, 2009. *Potensi Markisa di Kawasan Timur Indonesia*. Laporan Penelitian SADI-ACIAR (*Australian Centre for International Agriculture Research*).

Balestani, A.M., Moghaddam, P. A., Motlaq, A. M., Dolaty, H. 2012. *Sorting and Grading of Cherries on the Basis of Ripeness, Size and Defects by using Image Processing Techniques*. International Journal of Agriculture and Crop Sciences.

Bank Indonesia, 2008. *Sistem Informasi Pola Pembayaran/Lending Model Usaha Kecil – Budidaya Ma*rkisa, Jakarta.

Bautista, O.K. 1990. *Postharvest Technology for Southeast Asian Perishable Crops*. Technology and Livelifood Resource Centre. Makati, Metro Manila. Philippines

Belforte, G. dkk. 2014. *Soft Pneumatic Actuators for Rehabilitation. Actuators* 2014, *3*, 84-106; doi:10.3390/act3020084, ISSN 2076-0825.

News, http://berita2bahasa.com/berita/08/14180206-mentan-ng-ekspor-markisa-ditingkatkan [30 Mei 2017].

Budiharto, Dr. Widodo, S.Si., M. Kom., Purwanto, Ir. Djoko, M. Eng. 2015. *Robot Vision – Teknik Membangun Robot Cerdas Masa Depan (edisi revisi*). Penerbit Andi Yogyakarta.

Bonilla, Prieto dan Perez. 2017. *Mass and Volume Estimation of Passion Fruit using Digital Images.* IEEE Latin America Transactions.

Capiel. 1982. *Programmable Logic Controll*ers [Online]. Diakses http://www.capiel.eu [15 April 2017].

Devi, V., Vijayarekha, K. 2014. *Machine Vision Applications To Locate Fruits, Detect Defects And Remove Noise: A Review*. Rasayan Journal Vol. 7. No.1 Page 104-113.

Eissa, Ayman H. Amer & Khalik, Ayman A. Abdel. 2012. *Understanding Color Image Processing by Machine Vision for Biological Materials*. Intech.

Gill, Jasmeen, Sandhu, Dr. Parvinder Singh, Singh, Dr. Tejwant. 2014. *A Review of Automatic Fruit Classification using Soft Computing Techniques*. International Conference on Computer, Systems and Electronics Engineering (ICSCEE'2014) April 15-16, 2014 Johannesburg (South Africa).

Feng, Qingchun Feng, Wang, Xiaonan, Wang, Guohua, Li, Zhen Li. 2015. *Design and Test of Tomatoes Harvesting R*obot. Proceeding of the IEEE International Conference on Information and Automation.

Groover, Mikell P., Weiss, Mitchell, Nagel, Roger N., Odrey, Nicholas G. 1986. *Industrial Robotics Technology, Programming and Applications*. McGraww-Hill Book Company.

Hariyadi, Purwiyatno, Aini, Nur. 2015. *Dasar-Dasar Penanganan Pascapanen Buah dan Sayur*. Penerbit Alfabeta Bandung.

Iqbal, S. Md. dkk. 2015. *Estimation Of Size And Shape Of Citrus Fruits Using Image Processing For Automatic Grading*. 3rd International Conference on Signal Processing, Communication and Networking (ICSCN), IEEE.

Interlink Electronics*. FSR Force Sensing Resistor Integration Gude and Evaluation Parts Catalog*. http://www.interlinkelectronics.com/. [15 April].

Jyoti Jhawar. 2015. *Orange Sorting By Applying Pattern ognition On Colour Image.* International Conference on Information urity & Privacy (ICISP2015).

Jhuria, Monika, Kumar, Ashwani dan Borse, Rushikesh. 2013. *Image Processing For Smart Farming: Detection Disease And Fruit Grading*. Proceedings of the 2013 IEEE Second International Conference on Image Information Processing (ICIIP-2013).

Lu, Siyuan, Lu, Zhihai, dkk. 2016. *Fruit Classification by HPA-SLFN*. IEEE.

Londhe, D., Nawalade, S. 2013. *Grader: A review of different methods of grading for fruits and vegetables*. Agric. Eng. Int: CIGR Journal Vol. 15, No.3.

Mahendran R, Jayashree GC, Alagusundaram K. 2011**.** *Application of Computer Vision Technique on Sorting and Grading of Fruits and Vegetables*. Food Processing & Technology ISSNL 2157-7110.

Mhaski, Ruchita R. dkk. 2015. *Determination Of Ripeness And Grading Of Tomato Using Image Analysis On Raspberry Pi*. International Conference on Communication, Control and Intelligent Systems (CCIS).

Mishra, A., Astahana, P., Khanna, P. 2014. *The Quality Identification of fruits in Image Processing using* MATLAB. IJRET: International Journal of Research in Engineering and Technology Volume 03 Special Issue: 10.

Munir, R. 2004. *Pengolahan Citra Digital dengan Pendekatan Algoritmik. Bandung*. Penerbit Infomatika Bandung.

Najjari, B. dkk. 2012. *Modelling and Controller Design of Electro-Pneumatic Actuator Based on PWM*. International Journal of Robotics and Automation (IJRA), Vol. 1, No. 3, September 2012, pp. 125~136 ISSN: 2089-4856 _ 125.

Nanda, M. Achirul, Seminar, Boro K., Nandika, D. dan Maddu, A. 2018. *A Comparison Study of Kernel Functions in the Support Vector Machine and Its Application for Termite Detection*. Information, vol. 9, no. 1, p. 5/

Nandi, Chandra Sekhar, Tudu, Bipan Tudu dan Koley, Chiranjib. 2014. *Machine Vision Based Automatic Fruit Grading System using Fuzzy Algorithm*. 2014 International Conference on Control, Instrumentation, Energy & Communication(CIEC).

M. dkk., 2014. *Prototipe Alat Pendeteksian Kematangan Buah ng Belanda (Chypomandra betacca) Berdasarkan Warna ggunakan Mikrokontroller ATMega328*. Simposium Nasional ologi Terapan (SNTT)2 2014.

Nurdin, 2010. *Simulasi Sistem Kontrol Pengendalian Mesin Sortasi Otomatis untuk Buah Manggis dengan Menggunakan Bahasa Pemrograman Microsoft Visual Basic 6.0*. Jurnal Teknologi, Vol. 10, No. 1, April 2010: 14-19.

Nurtanio, I., dkk. 2013. *Classifying Cyst and Tumor Lesion Using Support Vector Machine Based on Dental Panoramic Images Texture Features*. p.9.

N., Oktaviano Yudha, dkk. 2011. *Aplikasi Komputer Vision untuk Identifikasi Kematangan Jeruk Nipis*. Tugas Akhir Mahasiswa Jurusan Teknik Elektro FTI-ITS.

Oladapo, Bankole I. dkk. 2016. *Model Design And Simulation of Automatic Sorting Machine Using Proximity Sensor*. Engineering Science ang Technology, An International Journal 19 (2016) 1452-1458.

Pandey, R., Naik, S., Marfatia, R., 2013. *Image Processing and Machine Learning for Automated Fruit Grading System: A Technical Review.* International Journal of Computer Application (0975-8887) Vol. 81 – No. 16.

Peteris, Eizentals Peteris, Koichi, Oka. 2015. *Green Pepper Stem Position Detection by using A Piezo Sensor*. IEEE

Pourdarbani, Razieh Pourdarbani dkk. 2015. *Study On An Automatic Sorting System For Date Fruits.* Journal of the Saudi Society of Agricultural Sciences (2015) 14, 83-90.

Prasetyo, Eko. 2014. *Data Mining Processing Data into Information using MATLAB*. First Edition, Yogyakarta, CV. ANDI OFFSET.

Putra, Darma, 2010. *Pengolahan Citra Digital*. Penerbit CV. Andi Offset, Yogyakarta.

Qiayi, Li dkk. 2014. *Study on Color Analyzer based on the Multiplexing of* TCS3200 *Color Sensor and Microcontroller*. International Journal of Hybrid Information Technology Vol. 7. No. 5 (2014), pp. 167-174.

Ramprabhu, J., Nandhini,S. 2014. *Enhanced Technique for Sorting and Grading The Fruit Quality using MSP430 Controller.* International Journal of Advances in Engineering & Technologies, Nov., 2014.

bhu, J., Nandhini,S. 2015. *Embedded Based System for The Fruit ity Management Using PIC Micro Controller*. International Journal ngineering and Computer Science Vol, 4 Issue 1 January 2015, No. 10051-10056.

Reddy, D., V., K., 2014. *Sorting of Objects Based on Colour by Pick and Place Robotic Arm and With Convenyor Belt Arrangement*. International Journal of Mechanical Engineering and Robotics Research.

Robinson, Ryan M., Kothera, Curt S. & Wereley, Norman M. 2014. *Control of a Heavy-Lift Robotic Manipulator with Pneumatic Arificial Muscles.* Actuators ISSN 2076-0825.

Rokunuzzaman, Md. Dkk. 2013. *Development of a low cost machine vision system for sorting of tomatoes*. Agric. Eng. Int: CIGR Journal Vol. 15, No.1.

Rukmana, 2003. *Usaha Tani Markisa*. Kanisius. Yogyakarta.

Sinar Tani. *Markisa Dataran Rendah untuk Ekspor*. http://tabloidsinartani.com/content/read/markisa-dataran-rendah-untuk-ekspor/. [30 Mei 2017]

Situmorang, M., 2013. *Pengenalan Komponen Warna Menggunakan Sensor Warna DT-Sense Berbasis Mikrokontroller ATMega 8535*. Prosiding Semirata FMIPA Universitas Lampung.

Sofu, M.M. dkk. 2016. *Design Of An Automatic Apple Sorting System Using Machine Vision*. Computers and Electronics in Agriculture 127 (2016) 395-405.

Suyanto. 2014. *Artificial Intelligence- Searching, Reasoning, Planning, Learning*. Penerbit Informatika. Bandung.

Technical Services Department Hunter Associates Laboratory, Inc.2008. *Hunter L, a, b Color Scale*, vol. 8, no. 9, p. 1, 2008.

Thiang, Indrotanoto, Leonardus, 2008. *Otomasi Pemisah Buah Tomat Berdasarkan Ukuran dan Warna Menggunakan Webcam Sebagai Sensor*. Seminar Nasional Ilmu Komputer dan Aplikasinya – SNIKA 2008.

Tho, Tuong Phuac, Thinh, Nguyen Truong, Bich, Nguyen Huy Bich. 2016. *Design and Development of the vision sorting system*. 3rd International Conference on Green Technology and Sustainable Development.

Timur. Yayasan Kalla Ekspor Markisa Jeneponto ke Singapura. /makassar.tribunnews.com/2013/12/15/yayasan-kalla-ekspor-isa-jeneponto-ke-singapura. [30 Mei 2017]

Wakhidah, N. 2010. *K-Means Algorithm Clustering*. *J. Transform.*, vol. 8, no. 1, pp. 33–39.

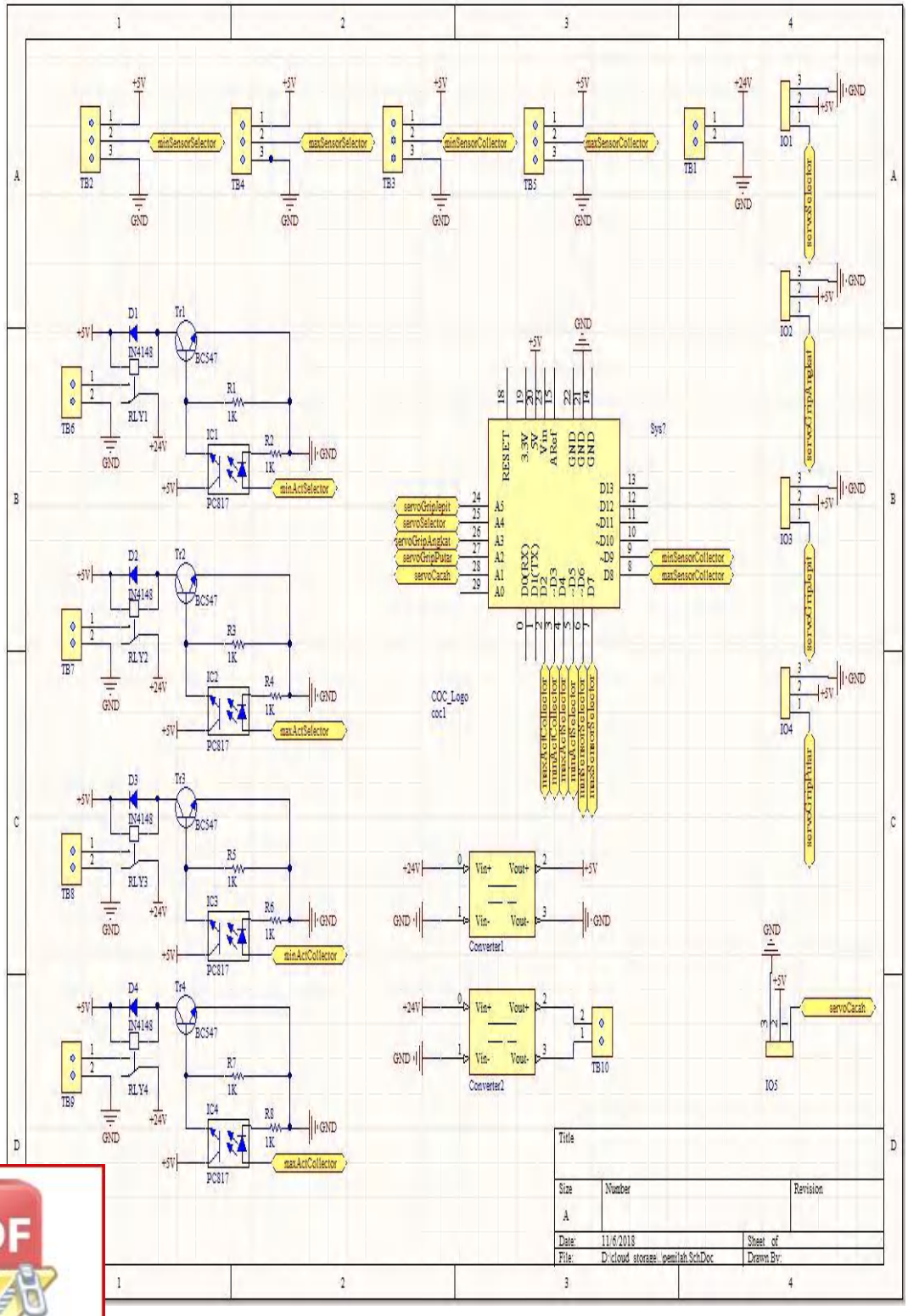
Wati, Dwi Ana Ratna. 2011. *Sistem Kendali Cerdas*: *Fuzzy Logic Controller (FLC), Jaringan Saraf Tiruan (JST), Algoritma Genetik (AG) dan Algoritma Particle Swarm Optimization (PSO)*. Graha Ilmu, Yogyakarta.

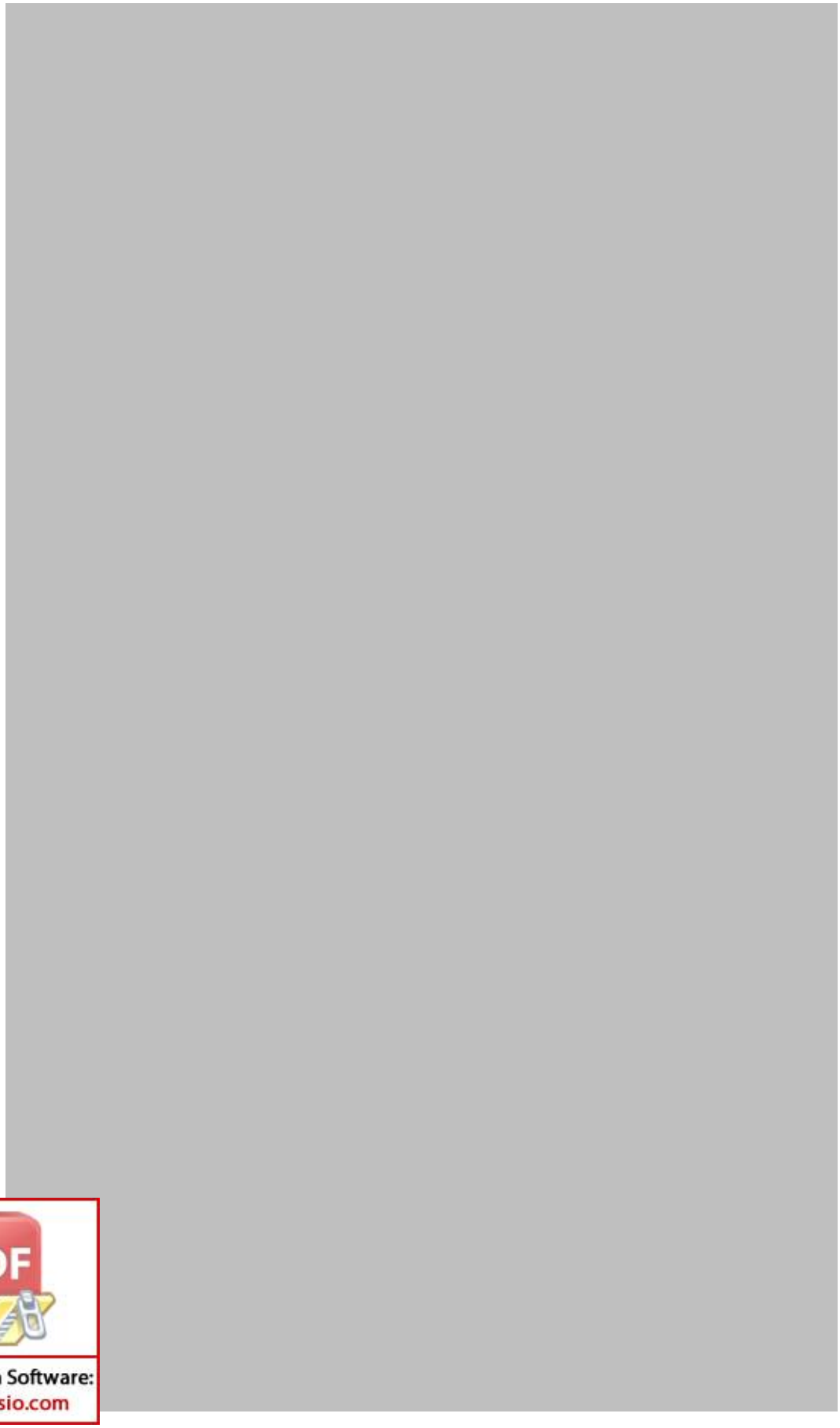
Yaguchi dkk. 2016. *Development of An Autonomous Tomato Harvesting Robot with Rotational Plucking Gripper*. RSJ International Conference on Intelligent Robots and Systems (IROS) Daejeon Convention Center.

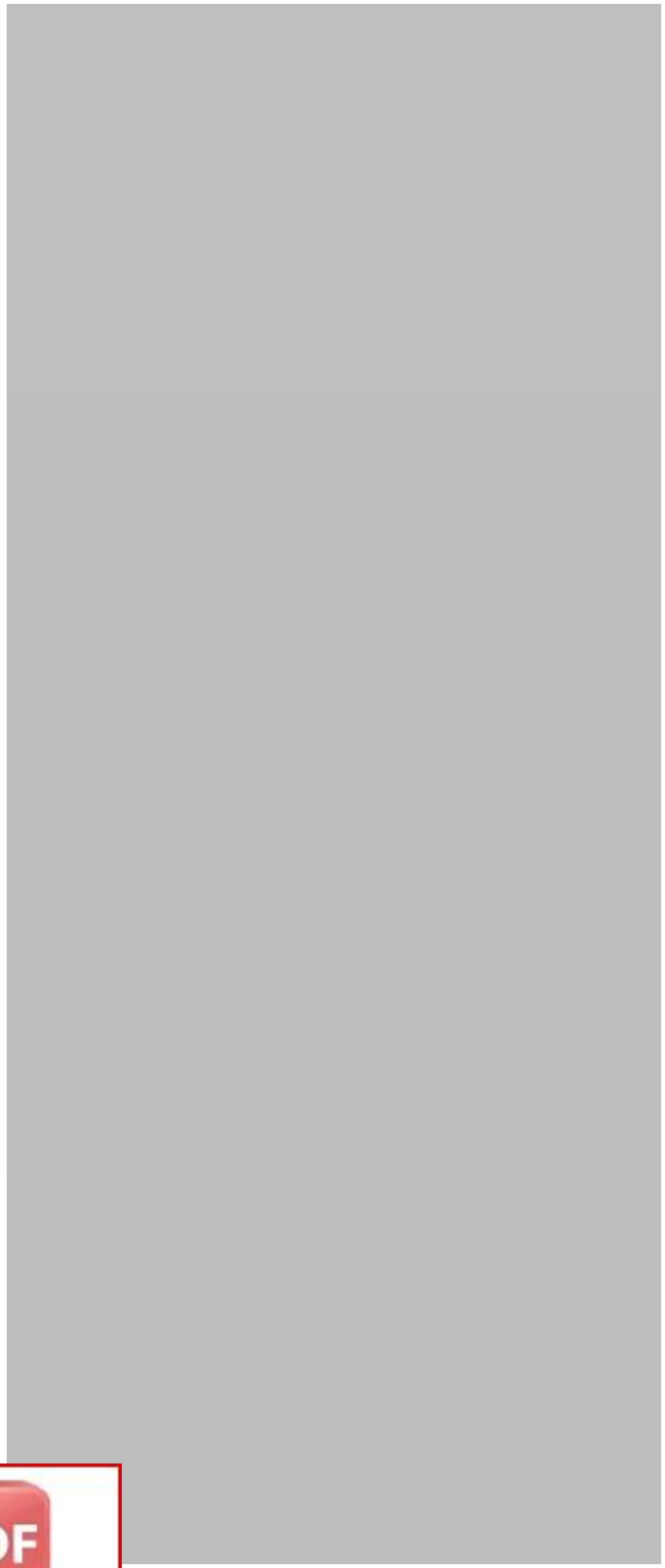
Zhao, Jie dkk. 2015. *Position Control of a Pneumatic Muscle Actuator Using RBF Neural Network Tuned PID Controller*. Mathematical Problems in Engineering, Volume 2015, Article ID 810231, 16 pages, http://dx.doi.org/10.1155/2015/810231.

LAMPIRAN

A. SCHEMATIC MESIN PEMILAH BUAH MARKISA







MATRIKS HASIL TRAINING MSVM PADA MATLAB

No	Matang	Mengkal	Mentah
1	-1.0002	-1.0755	-1.0000
2	-0.9996	-1.0000	-1.0001
3	-0.9998	-0.9998	-0.9998
4	-1.0530	-1.0004	-1.2298
5	-1.1427	-1.1363	-1.0268
6	-1.0001	-0.9997	-1.0003
7	-1.3795	-1.2964	-1.1610
8	-0.9994	-0.9998	-0.9998
9	-1.2580	-1.1911	-1.0760
10	-1.2196	-1.1650	-1.1533
11	-0.9996	-1.0001	-0.9997
12	-0.9999	-0.9999	-0.9997
13	-1.2102	-1.1683	-1.0535
14	-0.9997	-1.0003	-0.9997
15	-0.9999	-0.9999	-0.9998
16	-1.0004	-1.0003	-0.9999
17	-0.9996	-1.0019	-1.0009
18	-0.9998	-1.0000	-0.9998
19	-1.0003	-0.9999	-1.0400
20	-1.0897	-1.1237	-1.0515
21	-0.9997	-0.9994	-1.0002
22	-1.3557	-1.2740	-1.1298
23	-1.0748	-1.0792	-1.0667
24	-0.9992	-1.0000	-0.9999
25	-0.9994	-0.9999	-1.0003
26	-0.9998	-0.9998	-0.9999
27	-0.9998	-1.0001	-1.0003
28	-1.0001	-0.9998	-1.0001
29	-1.1892	-1.1400	-1.1293
30	-1.1007	-1.0927	-1.0801
31	1.0001	0.9998	-1.1916
32	1.0007	1.0000	-1.1040
33	1.0008	0.9997	-1.0000
34	1.0007	0.9999	-0.9998
35	1.0002	0.9996	-1.0001
36	1.0001	0.9999	-1.0000
37	1.1055	1.0003	-0.9998
38	1.0002	1.0000	-0.9998
39	1.2758	1.3186	-1.0681
40	1.0003	1.0001	-0.9999
41	1.0005	1.0000	-0.9996
42	1.1564	1.0390	-0.9999
	1.3956	1.5266	-1.2038
	1.0000	0.9999	-1.0002
	1.1187	1.1937	-1.1063
	1.0002	1.0001	-0.9998

47	1.0003	1.0000	-1.0000
48	1.4991	1.7256	-1.3492
49	1.0220	1.0224	-1.0277
50	1.0006	1.0005	-1.0000
51	1.0008	0.9997	-0.9999
52	1.0002	0.9999	-0.9999
53	1.0799	1.0935	-1.1094
54	0.9999	0.9998	-0.9998
55	1.1280	1.1605	-1.1011
56	1.0261	1.0066	-0.9999
57	1.0000	1.0001	-0.9998
58	0.9997	1.0001	-0.9999
59	1.0074	1.0003	-1.0000
60	1.0002	0.9995	-1.0002
61	1.1408	-1.2173	1.3428
62	1.0995	-1.0744	1.1496
63	1.0000	-1.0002	1.0002
64	1.0005	-0.9997	1.0001
65	1.0002	-1.0001	1.0000
66	1.0141	-0.9997	1.0004
67	1.0002	-1.0002	0.9999
68	1.1825	-1.2209	1.3712
69	1.0982	-1.2444	1.3120
70	1.2255	-1.2397	1.4299
71	1.1321	-1.2181	1.3434
72	1.1180	-1.0697	1.1488
73	1.0002	-0.9998	1.0001
74	1.0044	-1.0260	1.0365
75	1.1469	-1.0852	1.2107
76	1.0627	-1.0570	1.0960
77	1.0000	-1.0001	1.0000
78	1.0002	-0.9995	1.0001
79	0.9996	-1.0001	0.9996
80	0.9998	-1.0720	0.9998
81	1.0003	-1.0000	1.0004
82	1.1191	-1.1984	1.2792
83	0.9999	-1.0004	1.0001
84	1.1044	-1.1378	1.2194
85	0.9997	-0.9999	0.9997
86	1.0003	-1.0951	1.0647
87	1.0263	-1.0001	1.0131
88	1.2103	-1.1831	1.3676
89	1.1635	-1.1159	1.2549
90	1.1622	-1.1253	1.2829

MATRIKS HASIL TESTING MSVM PADA MATLAB

No	Matang	Mengkal	Mentah
1	-0.35297	-0.08142	-2.12247
2	-1.21092	-0.98334	-1.53287
3	-1.50556	-1.18125	-1.71516
4	-1.63858	-1.61328	-1.82788
5	-1.77996	-1.31518	-2.02256
6	0.001006	0.075268	-1.18327
7	-1.36016	-1.08513	-1.8007
8	-0.76757	-0.96744	-1.06806
9	-1.9693	-1.69055	-1.84637
10	-1.09156	-0.40096	-1.82088
11	3.012975	1.022005	-0.31955
12	0.300875	0.448543	-2.01394
13	0.050885	0.283081	-1.76057
14	1.831077	0.957462	-0.74998
15	1.014461	1.040118	-1.28
16	1.421888	1.522252	-1.52582
17	1.662041	1.083666	-1.03323
18	1.908603	1.201456	-1.29805
19	1.841302	1.42674	-1.63832
20	0.306263	0.495192	-2.11488
21	2.504119	-1.4763	1.507916
22	2.427836	-1.59973	1.414643
23	2.599134	-1.32047	1.183283
24	2.41543	-1.17121	1.317994
25	2.197881	-1.5959	1.454627
26	1.755436	-1.47175	1.420234
27	2.401092	-2.03333	1.749961
28	2.424658	-0.30194	0.5093
29	2.350075	-0.75785	0.579406
30	2.788748	-0.77118	0.806641

C. CODING MATLAB

C.1 SEGMENTASI FCM

```
clc; clear; close all;
save fitur_hsv_training
%memanggil frame
        files=dir('D:\Disertasi Terbaru\Disertasi\Coding
Markisa\Maret 19, 2018\SEGMENTASI FCM - Coba2\Training\frame
training 2\*.jpg');
        n =numel(files);
for i=1:n;
        str = strcat('D:\Disertasi Terbaru\Disertasi\Coding
Markisa\Maret 19, 2018\SEGMENTASI FCM - Coba2\Training\frame
training 2\', files(i).name);

        Img =imread(str);
    Img = imresize(Img,0.3);

    %Mengubah palet warna dari RGB menjadi L*a*b untuk segemntasi
menggunakan K-Means
            cform = makecform('srgb2lab'); %membuat struct
transformasi warna sesuai type di dlm kurung
            lab_markisa = applycform(Img,cform); %mengubah color
space Img sesuai type yg dipilih

            file_name=strcat('z1hasil lab_',num2str(i),'.jpg');
            imwrite(lab_markisa,file_name);

            %Memetakan nilai a*b* tiap-tiap piksel
            ab = double(lab_markisa(:,:,2:3)); %mengonversi ke
double nilai semua piksel di dimensi 2 (a) dan 3 (b)
            nrows = size(ab,1); %ukuran baris matriks ab
            ncols = size(ab,2); %ukuran kolom matriks ab
            ab = reshape(ab,nrows*ncols,2); %menjadikan matriks ab
menjadi ukuran nrows*ncols x 2

    nColors = 2;
    [centers,U] = fcm(ab,nColors);

    maxU = max(U);
    index1 = find(U(1,:) == maxU);
    index2 = find(U(2,:) == maxU);

cluster_idx = zeros(nrows,ncols);
cluster_idx(index1) = 1;
cluster_idx(index2) = 2;
pixel_labels = reshape(cluster_idx,nrows,ncols);

        abung tiap-tiap kluster yang terdapat markisa menjadi satu

        %markisa
```

```
            area_cluster1 = sum(sum(pixel_labels==1)) %jumlah dari
jumlah pixel_labels yang pikselnya bernilai 1 di setiap kolom
            area_cluster2 = sum(sum(pixel_labels==2))

            [~,cluster_markisa] =
min([area_cluster1,area_cluster2]); %cluster_markisa berisi posisi
baris dimana nilai minimum dari area_cluster1 dan 2
            markisa_bw = (pixel_labels==cluster_markisa);

            file_name=strcat('z2hasil
cluster_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

            %Mengisi lubang2 yang terdapat di dalam objek sehingga
menghasilkan objek
            %yang utuh
            markisa_bw = imfill(markisa_bw,'holes'); %mengisi
piksel yang tersambung tapi kosong
%
            file_name=strcat('z3setelah
imfill_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

            %Menghilangkan noise dengan menghapus objek yang area
nya berukuran kurang dari 3000
            if area_cluster1 >5000 && area_cluster2 >1000
            markisa_bw = bwareaopen(markisa_bw,1000);
            elseif area_cluster1 >5000 && area_cluster2 <1000
            markisa_bw = bwareaopen(markisa_bw,500);
            elseif area_cluster1 <5000
            markisa_bw = bwareaopen(markisa_bw,500);
            elseif area_cluster1 >50000 && area_cluster2 <500
            markisa_bw = bwareaopen(markisa_bw,1000);
            end

            file_name=strcat('z4setelah
hapusNOISE_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

        h = regionprops(markisa_bw, 'BoundingBox')
            if ~isempty(h)
            bb = h.BoundingBox;
                if length(h)== 1
                    if bb(1) > 1.00 && bb(2) > 10.00
                       jenis(i,:)=1;
                       s = h;
                       bb = s.BoundingBox;
                       markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
(4)),...

(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))

                    elseif bb(1) < 1.00 || bb(2) < 1.00
                        if bb(3) <300
                        jenis(i,:)=2;
```

```
                        bb = [173 108 56 56];
                        markisa_bw = imcrop (markisa_bw,bb);
                        else
                        bb = [173 100 56 56];
                        markisa_bw = imcrop (markisa_bw,bb);
                        end
                        end
                elseif length(h) > 1 && length(h) <5
                    jenis(i,:)=3;
                     s = h(2,:);
                     bb = s.BoundingBox;
                    if bb(3) > 200
                        jenis(i,:)=2;
                        bb = [173 108 56 56];
                        markisa_bw = imcrop (markisa_bw,bb);
                     else
                     markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
dingBox(4)),...

round(s(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))
);
                    end
                    elseif length(h) >= 5
                     jenis(i,:)=4;
                    s = h(4,:);
                      bb = s.BoundingBox;
                      markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
dingBox(4)),...

round(s(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))
);
                  end
              elseif isempty(h)
                  jenis(i,:)=5;
                  bb = [175 100 56 56];
             markisa_bw = imcrop (markisa_bw,bb);
             else
                 jenis(i,:)=6;
                  bb = [175 100 56 56];
                 markisa_bw = imcrop (markisa_bw,bb);
            end

            zzNILAI_H(i,:)=length(h);

            file_name=strcat('z5setelah
cropBBox_',num2str(i),'.bmp');
          imwrite(markisa_bw,file_name);
%

      %Membuat matriks 'Markisa' yang berisikan nilai Img
ntinya akan
      %digunakan untuk ekstraksi fitur warna

  Markisa = imcrop(Img,bb);
```

```
%Membuat Matriks R,G dan B
            R = Markisa(:,:,1); %memanggil matriks nilai R
            G = Markisa(:,:,2); %memanggil matriks nilai G
            B = Markisa(:,:,3); %memanggil matriks nilai B

            lab_markisa = applycform(Markisa,cform);
            ab = double(lab_markisa(:,:,2:3)); %mengonversi ke
double nilai semua piksel di dimensi 2 (a) dan 3 (b)
            nrows = size(ab,1); %ukuran baris matriks ab
            ncols = size(ab,2); %ukuran kolom matriks ab
            ab = reshape(ab,nrows*ncols,2);
            A = ab(:,1); %nilai A saja,

            %Inisialisasi matriks dengan nilai 0 ??
            R(~markisa_bw) = 0; %titik di R yang sama dgn
markisa_bw dan nilainya tidak ada atau 0, maka dijadikan 0
            G(~markisa_bw) = 0; %sama dgn R
            B(~markisa_bw) = 0; %sama dgn R

            markisa_rgb = cat(3,R,G,B); %menggabungkan array RGB
menjadi matriks 3 dimensi

            file_name=strcat('z6setelah jadi
rgb_',num2str(i),'.jpg');
            imwrite(markisa_rgb,file_name);

            dim = size(markisa_rgb);
            if dim(:,1) <= 70 && dim(:,2) > 70
                markisa_rgb = imcrop(markisa_rgb, [0 0 55 55]);
            elseif dim(:,1) > 70 && dim(:,1) <= 100
                markisa_rgb = imcrop(markisa_rgb, [20 12 60 60]);
            elseif dim(:,1) >= 100 && dim(:,2) > 150
                markisa_rgb = imcrop(markisa_rgb, [50 30 60 60]);
            end
            markisa_rgb = imresize(markisa_rgb, [56 56]);

            zzNILAI_DIM(i,:)=dim;

            file_name=strcat('z7setelah crop dan
resize_',num2str(i),'.jpg');
            imwrite(markisa_rgb,file_name);

            CiriR(i) = mean2(markisa_rgb(:,:,1)); %mencari rata-
rata Red di markisa_rgb
            CiriG(i) = mean2(markisa_rgb(:,:,2));
            CiriB(i) = mean2(markisa_rgb(:,:,3));
            CiriA(i) = mean2(A);
end

= [CiriR;CiriG;CiriB;CiriA];
= reshape(input1,[],90);

tur_rgb_training input1
```

C.2 SEGMENTASI K-MEANS

```
clc; clear; close all;

%memanggil frame
        files=dir('D:\Disertasi Terbaru\Disertasi\Coding
Markisa\Maret 19, 2018\SEGMENTASI KMEANS\training\frame
training\*.jpg');
        n =numel(files);
for i=1:n
        str = strcat('D:\Disertasi Terbaru\Disertasi\Coding
Markisa\Maret 19, 2018\SEGMENTASI KMEANS\training\frame
training\', files(i).name);

    Img =imread(str);
    Img = imresize(Img,0.3);

          %Mengubah palet warna dari RGB menjadi L*a*b untuk
segemntasi menggunakan K-Means
           cform = makecform('srgb2lab'); %membuat struct
transformasi warna sesuai type di dlm kurung
           lab_markisa = applycform(Img,cform); %mengubah color
space Img sesuai type yg dipilih

           file_name=strcat('z1hasil lab_',num2str(i),'.jpg');
           imwrite(lab_markisa,file_name);

           %Memetakan nilai a*b* tiap-tiap piksel
           ab = double(lab_markisa(:,:,2:3)); %mengonversi ke
double nilai semua piksel di dimensi 2 (a) dan 3 (b)
           nrows = size(ab,1); %ukuran baris matriks ab
           ncols = size(ab,2); %ukuran kolom matriks ab
           ab = reshape(ab,nrows*ncols,2); %menjadikan matriks ab
menjadi ukuran nrows*ncols x 2

           %Memilih jumlah kluster
           nColors = 2; %jumlah cluster  warna = 2
           %ulangi klustering sebanyak 3 kali untuk menghindari
lokal minima ??
           [cluster_idx,cluster_center] =
kmeans(ab,nColors,'distance','sqEuclidean','Replicates',3);
               %cluster_idx berisi indikasi cluster setiap titik.
               %jika ukuran matriks ab = n x p, maka ukuran idx
adalah n x 1
               %cluster_center berisi lokasi piksel centroid
setiap cluster
               %jika ukuran matriks ab = n x p dan k = jumlah
cluster, maka ukuran C adalah k x p

           data ab

           (cluster_idx==1,1),ab(cluster_idx==1,2),'r.','MarkerSize',1

           on
```

```
%
plot(ab(cluster_idx==2,1),ab(cluster_idx==2,2),'b.','MarkerSize',1
2)
% plot(cluster_center(:,1),cluster_center(:,2),'kx',...
%      'MarkerSize',12,'LineWidth',2)
% plot(cluster_center(:,1),cluster_center(:,2),'ko',...
%      'MarkerSize',12,'LineWidth',2)

            %Tandai tiap pixel pada citra dengan 'cluster_index'-
nya masing2.
            pixel_labels = reshape(cluster_idx,nrows,ncols);
                %menjadikan  matriks cluster_idx berukuran nrows x
ncols
                %sehingga setiap titik piksel memiliki nilai idx

%                 %Mengelompokkan tiap-tiap piksel ke dalam
kluster masing-masing
%             segmented_images = cell(1,3); %membuat cell array
berisi matriks ksong berukuran 1 x 3
%             rgb_label = repmat(pixel_labels, [1 1 3]);
%                 %membuat array rgb_label multidimensi terdiri
dari isi
%                 %matriks piksel_labels.
%                 %ukuran rgb_label = [ukuran baris piksel_labels
x 1, ukuran
%                 %kolom piksel_labels x 1, ukuran dimensi
piksel_labels x 3]
%
%             for k = 1:nColors %perulangan sebanyak 2 kali
%                 color = Img;    %membuat variabel color berisi
matriks Img
%                 color(rgb_label ~=k) =0; %semua titik di color
yang sama titiknya di rgb_label dan nilainya tidak sama dgn k,
diubah menajdi 0
%                 segmented_images{k} = color; %mengisi cell ke k
segmented images dgn matriks color
%
%             end

            %Object segmentation (tiap-tiap isi dari kluster
digabungkan di tiap2 kluster)
            area_cluster1 = sum(sum(pixel_labels==1)) %jumlah dari
jumlah pixel_labels yang pikselnya bernilai 1 di setiap kolom
            area_cluster2 = sum(sum(pixel_labels==2))   %jumlah
dari jumlah pixel_labels yang pikselnya bernilai 2 di setiap kolom

            zzNILAI_area_cluster1(i,:)=area_cluster1;
            zzNILAI_area_cluster2(i,:)=area_cluster2;

            %Menggabung tiap-tiap kluster yang terdapat markisa
        satu citra
            %markisa
            [~,cluster_markisa] =
        ea_cluster1,area_cluster2]); %cluster_markisa berisi posisi
        imana nilai minimum dari area_cluster1 dan 2
```

```
            markisa_bw = (pixel_labels==cluster_markisa);
%markisa_bw1 nilainya 1 jika titik piksel matriks pixel_labels
sama dengan cluster_markisa, jika tidak diberi nilai 0

            file_name=strcat('z2hasil
cluster_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

            %Mengisi lubang2 yang terdapat di dalam objek sehingga
menghasilkan objek
            %yang utuh
            markisa_bw = imfill(markisa_bw,'holes'); %mengisi
piksel yang tersambung tapi kosong
%
            file_name=strcat('z3setelah
imfill_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

            %Menghilangkan noise dengan menghapus objek yang area
nya berukuran kurang dari 3000
            if area_cluster1 >5000 && area_cluster2 >1000
            markisa_bw = bwareaopen(markisa_bw,1000);
            elseif area_cluster1 >5000 && area_cluster2 <1000
            markisa_bw = bwareaopen(markisa_bw,500);
            elseif area_cluster1 <5000
            markisa_bw = bwareaopen(markisa_bw,500);
            elseif area_cluster1 >50000 && area_cluster2 <500
            markisa_bw = bwareaopen(markisa_bw,1000);
            end

            file_name=strcat('z4setelah
hapusNOISE_',num2str(i),'.bmp');
            imwrite(markisa_bw,file_name);

           h = regionprops(markisa_bw, 'BoundingBox')
            if ~isempty(h)
            bb = h.BoundingBox;
                if length(h)== 1
                    if bb(1) > 1.00 && bb(2) > 10.00
                        jenis(i,:)=1;
                        s = h;
                        bb = s.BoundingBox;
                        markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
dingBox(4)),...

round(s(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))
);
                    elseif bb(1) < 1.00 || bb(2) < 1.00
                        if bb(3) <300
                        jenis(i,:)=2;
                        bb = [173 108 56 56];
                        markisa_bw = imcrop (markisa_bw,bb);
                        else
                        bb = [173 100 56 56];
                        markisa_bw = imcrop (markisa_bw,bb);
```

```
                    end
                end
            elseif length(h) > 1 && length(h) <5
                jenis(i,:)=3;
                s = h(2,:);
                bb = s.BoundingBox;
                if bb(3) > 200
                    jenis(i,:)=2;
                    bb = [173 108 56 56];
                    markisa_bw = imcrop (markisa_bw,bb);
                else
                markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
dingBox(4)),...

round(s(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))
);
                end
            elseif length(h) >= 5
                jenis(i,:)=4;
                s = h(4,:);
                bb = s.BoundingBox;
                markisa_bw =
markisa_bw(round(s(1).BoundingBox(2):s(1).BoundingBox(2)+s(1).Boun
dingBox(4)),...

round(s(1).BoundingBox(1):s(1).BoundingBox(1)+s(1).BoundingBox(3))
);
                end
        elseif isempty(h)
            jenis(i,:)=5;
            bb = [175 100 56 56];
            markisa_bw = imcrop (markisa_bw,bb);
        else
            jenis(i,:)=6;
            bb = [175 100 56 56];
            markisa_bw = imcrop (markisa_bw,bb);
        end

        zzNILAI_H(i,:)=length(h);

        file_name=strcat('z5setelah
cropBBox_',num2str(i),'.bmp');
        imwrite(markisa_bw,file_name);


        %Membuat matriks 'Markisa' yang berisikan nilai Img
yang nantinya akan
        %digunakan untuk ekstraksi fitur warna

        Markisa = imcrop(Img,bb);

        %Membuat Matriks R,G dan B
        R = Markisa(:,:,1); %memanggil matriks nilai R
        G = Markisa(:,:,2); %memanggil matriks nilai G
        B = Markisa(:,:,3); %memanggil matriks nilai B
```

```
            lab_markisa = applycform(Markisa,cform);
            ab = double(lab_markisa(:,:,2:3)); %mengonversi ke
double nilai semua piksel di dimensi 2 (a) dan 3 (b)
            nrows = size(ab,1); %ukuran baris matriks ab
            ncols = size(ab,2); %ukuran kolom matriks ab
            ab = reshape(ab,nrows*ncols,2);
            A = ab(:,1); %nilai A saja,

            %Inisialisasi matriks dengan nilai 0 ??
            R(~markisa_bw) = 0; %titik di R yang sama dgn
markisa_bw dan nilainya tidak ada atau 0, maka dijadikan 0
            G(~markisa_bw) = 0; %sama dgn R
            B(~markisa_bw) = 0; %sama dgn R

            markisa_rgb = cat(3,R,G,B); %menggabungkan array RGB
menjadi matriks 3 dimensi

            file_name=strcat('z6setelah jadi
rgb_',num2str(i),'.jpg');
            imwrite(markisa_rgb,file_name);

            dim = size(markisa_rgb);
            if dim(:,1) <= 70 && dim(:,2) > 70
                markisa_rgb = imcrop(markisa_rgb, [0 0 55 55]);
            elseif dim(:,1) > 70 && dim(:,1) <= 100
                markisa_rgb = imcrop(markisa_rgb, [20 12 60 60]);
            elseif dim(:,1) >= 100 && dim(:,2) > 150
                markisa_rgb = imcrop(markisa_rgb, [50 30 60 60]);
            end
            markisa_rgb = imresize(markisa_rgb, [56 56]);

            zzNILAI_DIM(i,:)=dim;

            file_name=strcat('z7setelah crop dan
resize_',num2str(i),'.jpg');
            imwrite(markisa_rgb,file_name);

            CiriR(i) = mean2(markisa_rgb(:,:,1)); %mencari rata-
rata Red di markisa_rgb
            CiriG(i) = mean2(markisa_rgb(:,:,2));
            CiriB(i) = mean2(markisa_rgb(:,:,3));
            CiriA(i) = mean2(A);
end

input1 = [CiriR;CiriG;CiriB;CiriA];
input = reshape(input1,[],90);

      fitur_rgba_training input
```

C.3 KLASIFIKASI DECISION TREE

```
clc; clear; close all; warning off all;
save TREE_FCM
%Training Session
load fitur_rgba_training;
t = length(input);
input = input';
target = zeros(1,90);
target(:,1:30) = 1;
target(:,31:60) = 2;
target(:,61:90) = 3;
target = target';
tree = classregtree(input, target);
view(tree);
save TREE_FCM tree -append
output = eval(tree,input);

hasil = confusionmat(target,output);
akurasi = sum(diag(hasil))/t*100
```

C.4 KLASIFIKASI NEURAL NETWORK

```
clc; clear; close all; warning off all;

save NET_FCM
%Training Session
load fitur_rgba_training
ndata=length(input);
target = zeros(1,90);
target(:,1:30) = 1;
target(:,31:60) = 2;
target(:,61:90) = 3;
net = newff(input,target,[50 10],{'tansig','tansig'},'trainscg');
net.trainParam.epochs = 1000;
net.trainParam.goal = 1e-3;
net.trainParam.lr = 0.01;
net.trainParam.lr_inc = 1.05;
net.trainParam.show = 25;
net.TrainParam.showWindow;
net.trainParam.time = inf;
net.trainParam.min_grad = 1e-10;
net.trainParam.max_fail = 100;

net_NEW= train(net,input,target);

output = round(sim(net_NEW,input));
save NET_FCM net_NEW -append
```


```
find(output==target);

= sum(m)/ndata*100;
```

C.5 KLASIFIKASI MULTI-CLASS SVM

```
clc
clear all

save MODEL_FCM
load fitur_rgba_training;
data_latih = input;
data_latih = data_latih';
%Ganti label kelas dengan matriks nomor indeks kelas
%load kelas;
kelas_1 = double(ismember(KelasTraining, 'Matang'));
kelas_2 = double(ismember(KelasTraining, 'Mengkal'));
kelas_3 = double(ismember(KelasTraining, 'Mentah'));

kelas_latih = [kelas_1 kelas_2 kelas_3];
kelas_latihT = kelas_latih';
%[a, kelas_latih] = max(kelas_latih');
[a, kelas_latih] = max(kelas_latihT);
kelas_latih = kelas_latih';


%membentuk klasifikator 1
kelas_latih_1_1 = kelas_1;
kelas_latih_1_0 = kelas_2 + kelas_3;
kelas_latih_1 = [kelas_latih_1_1 kelas_latih_1_0];
[a1, kelas_latih_1] = max(kelas_latih_1');
kelas_latih_1 = kelas_latih_1';
kelas_latih_1 = 2 - kelas_latih_1;
%pembangunan model klasifikasi
ModelSVM_1 = svmtrain(data_latih, kelas_latih_1,...
'BoxConstraint',5,'Kernel_Function','rbf','RBF_Sigma',5);
display('Klasifikator 1');


%membentuk klasifikator 2
kelas_latih_2_1 = kelas_1 + kelas_3;
kelas_latih_2_0 = kelas_2;
kelas_latih_2 = [kelas_latih_2_1 kelas_latih_2_0];
[a2, kelas_latih_2] = max(kelas_latih_2');
kelas_latih_2 = kelas_latih_2';
kelas_latih_2 = 2 - kelas_latih_2;
ModelSVM_2 = svmtrain(data_latih, kelas_latih_2,...
'BoxConstraint',5,'Kernel_Function','rbf','RBF_Sigma',5);
display('Klasifikator 2');

%membentuk klasifikator 3
kelas_latih_3_1 = kelas_1 + kelas_2;
kelas_latih_3_0 = kelas_3;
atih_3 = [kelas_latih_3_1 kelas_latih_3_0];
las_latih_3] = max(kelas_latih_3');
atih_3 = kelas_latih_3';
atih_3 = 2 - kelas_latih_3;
gunan model klasifikasi
M_3 = svmtrain(data_latih, kelas_latih_3,...
```

```
'BoxConstraint',5,'Kernel_Function','rbf','RBF_Sigma',5);
display('Klasifikator 3');

model1_kmean = ModelSVM_1;
model2_kmean = ModelSVM_2;
model3_kmean = ModelSVM_3;

save MODEL_FCM ModelSVM_1 -append
save MODEL_FCM ModelSVM_2 -append
save MODEL_FCM ModelSVM_3 -append

%MELAKUKAN PREDIKSI
kelas_1 = [1 1 1];
kelas_2 = [0 0 1];
kelas_3 = [0 1 0];

n=90;
for i=1:n
    prediksi(1) = svmclassify(ModelSVM_1, data_latih(i,:));
    prediksi(2) = svmclassify(ModelSVM_2, data_latih(i,:));
    prediksi(3) = svmclassify(ModelSVM_3, data_latih(i,:));
    jarak(1) = sum(xor(prediksi, kelas_1));
    jarak(2) = sum(xor(prediksi, kelas_2));
    jarak(3) = sum(xor(prediksi, kelas_3));
    [a, idx_kelas] = min(jarak);
    kelas_uji_hasil(i) = idx_kelas;
end
kelas_uji_hasil = kelas_uji_hasil';
hasil_prediksi = confusionmat(kelas_latih, kelas_uji_hasil)
akurasi = sum(diag(hasil_prediksi))/n*100
```

D. CODING PYTHON

```
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtCore import QDir, Qt
from PyQt5.QtWidgets import QGraphicsScene, QFileDialog,
QMessageBox
from PyQt5.QtGui import QPixmap, QImage
import sys
import cv2
import numpy as np
import queue
import math
import MySQLdb
from copy import deepcopy
from sklearn.svm import SVC
from sklearn.cluster import KMeans
import time as tm
from time import time
serial


c
dbx
cap
ard
```

```
programSetup = True
programRunning = True

port = '/dev/ttyACM0'

ard = serial.Serial(port,9600,timeout=0.01)

def kmeans(x):
    img = cv2.cvtColor(x, cv2.COLOR_BGR2LAB)
    channel = cv2.split(img)
    channelIndices = 1
    image = img[:,:,channelIndices]
    if len(image.shape) == 2:
        image.reshape(image.shape[0], image.shape[1],1)
    reshaped = image.reshape(image.shape[0] * image.shape[1], 1)
    cluster = 2
    kmeans = KMeans(n_clusters=cluster, n_init=3,
max_iter=10).fit(reshaped)
    h,w = reshaped.shape[:2]

    width = np.array(kmeans.labels_, dtype=np.uint8)
    high = (h,w)

    clustering = np.reshape(np.array(kmeans.labels_,
dtype=np.uint8),
                                        (image.shape[0],
image.shape[1]))

    sortedLabels = sorted([n for n in range(cluster)],
                              key=lambda x: -np.sum(clustering ==
x))

    kmeansImage = np.zeros(image.shape[:2], dtype=np.uint8)

    for i, label in enumerate(sortedLabels):
        kmeansImage[clustering == label] = int((255) / (cluster -
1)) * i

    imContour, contours, hierarchy = cv2.findContours(kmeansImage,
cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
    mask = np.zeros(image.shape[:2], dtype=np.uint8)
    if (len(contours))>0:
        areaMax = 0
        i = contours[0]
        savedContours = 0
        for i in range (len(contours)):
            area = cv2.contourArea(contours[i])
            if area> areaMax:
                areaMax = area
                savedContours = i

        #cv2.drawContours(mask,  contours, savedContours,
,255),-1)
        (xi,yi),radius =
nclosingCircle(contours[savedContours])
        center = (int(xi),int(yi))
        radius = int(radius)
```

```
            cv2.circle(mask,center,radius,(255,255,255),-1)

        b,g,r,c = cv2.mean(x,mask)
        crop_lab = cv2.cvtColor(x, cv2.COLOR_BGR2LAB)
        L, a, q, d = cv2.mean(crop_lab,mask)
        #cv2.drawContours(x,  contours, savedContours, (100,255,50),1)
        cv2.circle(x,center,radius,(255,255,0),3)

        return x,b,g,r,a


class Ui_Form(QtWidgets.QWidget):
    def __init__(self):
        QtWidgets.QWidget.__init__(self)
        self.setupUi(self)
    def setupUi(self, Form):
        Form.setObjectName("Form")
        Form.resize(1065, 788)
        palette = QtGui.QPalette()
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.WindowText, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Button, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Light, brush)
        brush = QtGui.QBrush(QtGui.QColor(212, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Midlight, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Dark, brush)
        brush = QtGui.QBrush(QtGui.QColor(113, 170, 170))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Mid, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Text, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
        Palette.BrightText, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
        Palette.ButtonText, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
```

```
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Base, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Window, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.Shadow, brush)
        brush = QtGui.QBrush(QtGui.QColor(212, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.AlternateBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 220))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.ToolTipBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.ToolTipText, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.WindowText, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Button, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Light, brush)
        brush = QtGui.QBrush(QtGui.QColor(212, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Midlight, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Dark, brush)
        brush = QtGui.QBrush(QtGui.QColor(113, 170, 170))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Mid, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Text, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
Palette.BrightText, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
```

```
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.ButtonText, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Base, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Window, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.Shadow, brush)
        brush = QtGui.QBrush(QtGui.QColor(212, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.AlternateBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 220))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.ToolTipBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.ToolTipText, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.WindowText, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Button, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Light, brush)
        brush = QtGui.QBrush(QtGui.QColor(212, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Midlight, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Dark, brush)
        brush = QtGui.QBrush(QtGui.QColor(113, 170, 170))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Mid, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
Palette.Text, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
```

```
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.BrightText, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.ButtonText, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Base, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Window, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.Shadow, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 255, 255))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.AlternateBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(255, 255, 220))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.ToolTipBase, brush)
        brush = QtGui.QBrush(QtGui.QColor(0, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.ToolTipText, brush)
        Form.setPalette(palette)
        self.gvLiveCam = QtWidgets.QGraphicsView(Form)
        self.gvLiveCam.setGeometry(QtCore.QRect(20, 180, 501,
341))
        self.gvLiveCam.setObjectName("gvLiveCam")
        self.gvSisi1 = QtWidgets.QGraphicsView(Form)
        self.gvSisi1.setGeometry(QtCore.QRect(550, 90, 121, 101))
        self.gvSisi1.setObjectName("gvSisi1")
        self.label = QtWidgets.QLabel(Form)
        self.label.setGeometry(QtCore.QRect(540, 200, 47, 13))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(Form)
        self.label_2.setGeometry(QtCore.QRect(540, 230, 47, 13))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(Form)
        self.label_3.setGeometry(QtCore.QRect(540, 260, 47, 13))
        self.label_3.setObjectName("label_3")
        self.label_4 = QtWidgets.QLabel(Form)
        self.label_4.setGeometry(QtCore.QRect(540, 290, 47, 13))
        self.label_4.setObjectName("label_4")
        self.teR1 = QtWidgets.QTextEdit(Form)
        self.teR1.setGeometry(QtCore.QRect(560, 200, 104, 21))
        self.teR1.setObjectName("teR1")
        self.teG1 = QtWidgets.QTextEdit(Form)
        self.teG1.setGeometry(QtCore.QRect(560, 230, 104, 21))
        self.teG1.setObjectName("teG1")
        self.teB1 = QtWidgets.QTextEdit(Form)
```

```
self.teB1.setGeometry(QtCore.QRect(560, 260, 104, 21))
self.teB1.setObjectName("teB1")
self.teA1 = QtWidgets.QTextEdit(Form)
self.teA1.setGeometry(QtCore.QRect(560, 290, 104, 21))
self.teA1.setObjectName("teA1")
self.label_5 = QtWidgets.QLabel(Form)
self.label_5.setGeometry(QtCore.QRect(730, 260, 47, 13))
self.label_5.setObjectName("label_5")
self.teB2 = QtWidgets.QTextEdit(Form)
self.teB2.setGeometry(QtCore.QRect(750, 260, 104, 21))
self.teB2.setObjectName("teB2")
self.label_6 = QtWidgets.QLabel(Form)
self.label_6.setGeometry(QtCore.QRect(730, 290, 47, 13))
self.label_6.setObjectName("label_6")
self.gvSisi2 = QtWidgets.QGraphicsView(Form)
self.gvSisi2.setGeometry(QtCore.QRect(740, 90, 121, 101))
self.gvSisi2.setObjectName("gvSisi2")
self.label_7 = QtWidgets.QLabel(Form)
self.label_7.setGeometry(QtCore.QRect(730, 200, 47, 13))
self.label_7.setObjectName("label_7")
self.teG2 = QtWidgets.QTextEdit(Form)
self.teG2.setGeometry(QtCore.QRect(750, 230, 104, 21))
self.teG2.setObjectName("teG2")
self.teR2 = QtWidgets.QTextEdit(Form)
self.teR2.setGeometry(QtCore.QRect(750, 200, 104, 21))
self.teR2.setObjectName("teR2")
self.label_8 = QtWidgets.QLabel(Form)
self.label_8.setGeometry(QtCore.QRect(730, 230, 47, 13))
self.label_8.setObjectName("label_8")
self.teA2 = QtWidgets.QTextEdit(Form)
self.teA2.setGeometry(QtCore.QRect(750, 290, 104, 21))
self.teA2.setObjectName("teA2")
self.gvSisi3 = QtWidgets.QGraphicsView(Form)
self.gvSisi3.setGeometry(QtCore.QRect(920, 90, 121, 101))
self.gvSisi3.setObjectName("gvSisi3")
self.teR3 = QtWidgets.QTextEdit(Form)
self.teR3.setGeometry(QtCore.QRect(930, 200, 104, 21))
self.teR3.setObjectName("teR3")
self.teB3 = QtWidgets.QTextEdit(Form)
self.teB3.setGeometry(QtCore.QRect(930, 260, 104, 21))
self.teB3.setObjectName("teB3")
self.teA3 = QtWidgets.QTextEdit(Form)
self.teA3.setGeometry(QtCore.QRect(930, 290, 104, 21))
self.teA3.setObjectName("teA3")
self.label_9 = QtWidgets.QLabel(Form)
self.label_9.setGeometry(QtCore.QRect(910, 230, 47, 13))
self.label_9.setObjectName("label_9")
self.label_10 = QtWidgets.QLabel(Form)
self.label_10.setGeometry(QtCore.QRect(910, 290, 47, 13))
self.label_10.setObjectName("label_10")
self.label_11 = QtWidgets.QLabel(Form)
self.label_11.setGeometry(QtCore.QRect(910, 200, 47, 13))
self.label_11.setObjectName("label_11")
self.teG3 = QtWidgets.QTextEdit(Form)
self.teG3.setGeometry(QtCore.QRect(930, 230, 104, 21))
self.teG3.setObjectName("teG3")
self.label_12 = QtWidgets.QLabel(Form)
```

```
self.label_12.setGeometry(QtCore.QRect(910, 260, 47, 13))
self.label_12.setObjectName("label_12")
self.teG4 = QtWidgets.QTextEdit(Form)
self.teG4.setGeometry(QtCore.QRect(560, 480, 104, 21))
self.teG4.setObjectName("teG4")
self.teR5 = QtWidgets.QTextEdit(Form)
self.teR5.setGeometry(QtCore.QRect(750, 450, 104, 21))
self.teR5.setObjectName("teR5")
self.teB4 = QtWidgets.QTextEdit(Form)
self.teB4.setGeometry(QtCore.QRect(560, 510, 104, 21))
self.teB4.setObjectName("teB4")
self.label_13 = QtWidgets.QLabel(Form)
self.label_13.setGeometry(QtCore.QRect(540, 450, 47, 13))
self.label_13.setObjectName("label_13")
self.gvSisi4 = QtWidgets.QGraphicsView(Form)
self.gvSisi4.setGeometry(QtCore.QRect(550, 340, 121, 101))
self.gvSisi4.setObjectName("gvSisi4")
self.teG6 = QtWidgets.QTextEdit(Form)
self.teG6.setGeometry(QtCore.QRect(930, 480, 104, 21))
self.teG6.setObjectName("teG6")
self.teR4 = QtWidgets.QTextEdit(Form)
self.teR4.setGeometry(QtCore.QRect(560, 450, 104, 21))
self.teR4.setObjectName("teR4")
self.label_14 = QtWidgets.QLabel(Form)
self.label_14.setGeometry(QtCore.QRect(910, 450, 47, 13))
self.label_14.setObjectName("label_14")
self.teB5 = QtWidgets.QTextEdit(Form)
self.teB5.setGeometry(QtCore.QRect(750, 510, 104, 21))
self.teB5.setObjectName("teB5")
self.label_15 = QtWidgets.QLabel(Form)
self.label_15.setGeometry(QtCore.QRect(540, 510, 47, 13))
self.label_15.setObjectName("label_15")
self.gvSisi5 = QtWidgets.QGraphicsView(Form)
self.gvSisi5.setGeometry(QtCore.QRect(740, 340, 121, 101))
self.gvSisi5.setObjectName("gvSisi5")
self.gvSisi6 = QtWidgets.QGraphicsView(Form)
self.gvSisi6.setGeometry(QtCore.QRect(920, 340, 121, 101))
self.gvSisi6.setObjectName("gvSisi6")
self.label_16 = QtWidgets.QLabel(Form)
self.label_16.setGeometry(QtCore.QRect(910, 510, 47, 13))
self.label_16.setObjectName("label_16")
self.label_17 = QtWidgets.QLabel(Form)
self.label_17.setGeometry(QtCore.QRect(730, 450, 47, 13))
self.label_17.setObjectName("label_17")
self.teA5 = QtWidgets.QTextEdit(Form)
self.teA5.setGeometry(QtCore.QRect(750, 540, 104, 21))
self.teA5.setObjectName("teA5")
self.label_18 = QtWidgets.QLabel(Form)
self.label_18.setGeometry(QtCore.QRect(910, 480, 47, 13))
self.label_18.setObjectName("label_18")
self.label_19 = QtWidgets.QLabel(Form)
self.label_19.setGeometry(QtCore.QRect(910, 540, 47, 13))
self.label_19.setObjectName("label_19")
self.label_20 = QtWidgets.QLabel(Form)
self.label_20.setGeometry(QtCore.QRect(730, 480, 47, 13))
self.label_20.setObjectName("label_20")
self.label_21 = QtWidgets.QLabel(Form)
```

```
        self.label_21.setGeometry(QtCore.QRect(730, 540, 47, 13))
        self.label_21.setObjectName("label_21")
        self.teA6 = QtWidgets.QTextEdit(Form)
        self.teA6.setGeometry(QtCore.QRect(930, 540, 104, 21))
        self.teA6.setObjectName("teA6")
        self.label_22 = QtWidgets.QLabel(Form)
        self.label_22.setGeometry(QtCore.QRect(540, 540, 47, 13))
        self.label_22.setObjectName("label_22")
        self.teG5 = QtWidgets.QTextEdit(Form)
        self.teG5.setGeometry(QtCore.QRect(750, 480, 104, 21))
        self.teG5.setObjectName("teG5")
        self.label_23 = QtWidgets.QLabel(Form)
        self.label_23.setGeometry(QtCore.QRect(730, 510, 47, 13))
        self.label_23.setObjectName("label_23")
        self.teR6 = QtWidgets.QTextEdit(Form)
        self.teR6.setGeometry(QtCore.QRect(930, 450, 104, 21))
        self.teR6.setObjectName("teR6")
        self.teA4 = QtWidgets.QTextEdit(Form)
        self.teA4.setGeometry(QtCore.QRect(560, 540, 104, 21))
        self.teA4.setObjectName("teA4")
        self.teB6 = QtWidgets.QTextEdit(Form)
        self.teB6.setGeometry(QtCore.QRect(930, 510, 104, 21))
        self.teB6.setObjectName("teB6")
        self.label_24 = QtWidgets.QLabel(Form)
        self.label_24.setGeometry(QtCore.QRect(540, 480, 47, 13))
        self.label_24.setObjectName("label_24")
        self.pbStart = QtWidgets.QPushButton(Form)
        self.pbStart.setGeometry(QtCore.QRect(80, 80, 121, 21))
        self.pbStart.setObjectName("pbStart")
        self.pbStop = QtWidgets.QPushButton(Form)
        self.pbStop.setGeometry(QtCore.QRect(210, 80, 121, 21))
        self.pbStop.setObjectName("pbStop")
        self.label_25 = QtWidgets.QLabel(Form)
        self.label_25.setGeometry(QtCore.QRect(0, 20, 1061, 31))
        font = QtGui.QFont()
        font.setPointSize(20)
        font.setBold(True)
        font.setWeight(75)
        self.label_25.setFont(font)
        self.label_25.setAlignment(QtCore.Qt.AlignCenter)
        self.label_25.setObjectName("label_25")
        self.pbKalibrasi = QtWidgets.QPushButton(Form)
        self.pbKalibrasi.setGeometry(QtCore.QRect(340, 80, 121,
21))
        self.pbKalibrasi.setObjectName("pbKalibrasi")
        self.label_32 = QtWidgets.QLabel(Form)
        self.label_32.setGeometry(QtCore.QRect(20, 140, 491, 31))
        font = QtGui.QFont()
        font.setPointSize(20)
        font.setBold(True)
        font.setWeight(75)
        self.label_32.setFont(font)
        self.label_32.setAlignment(QtCore.Qt.AlignCenter)
        self.label_32.setObjectName("label_32")
        self.label_33 = QtWidgets.QLabel(Form)
        self.label_33.setGeometry(QtCore.QRect(550, 70, 111, 16))
        self.label_33.setAlignment(QtCore.Qt.AlignCenter)
```

```
        self.label_33.setObjectName("label_33")
        self.label_34 = QtWidgets.QLabel(Form)
        self.label_34.setGeometry(QtCore.QRect(740, 70, 111, 16))
        self.label_34.setAlignment(QtCore.Qt.AlignCenter)
        self.label_34.setObjectName("label_34")
        self.label_35 = QtWidgets.QLabel(Form)
        self.label_35.setGeometry(QtCore.QRect(920, 70, 111, 16))
        self.label_35.setAlignment(QtCore.Qt.AlignCenter)
        self.label_35.setObjectName("label_35")
        self.label_36 = QtWidgets.QLabel(Form)
        self.label_36.setGeometry(QtCore.QRect(550, 320, 111, 16))
        self.label_36.setAlignment(QtCore.Qt.AlignCenter)
        self.label_36.setObjectName("label_36")
        self.label_37 = QtWidgets.QLabel(Form)
        self.label_37.setGeometry(QtCore.QRect(750, 320, 111, 16))
        self.label_37.setAlignment(QtCore.Qt.AlignCenter)
        self.label_37.setObjectName("label_37")
        self.label_38 = QtWidgets.QLabel(Form)
        self.label_38.setGeometry(QtCore.QRect(920, 320, 111, 16))
        self.label_38.setAlignment(QtCore.Qt.AlignCenter)
        self.label_38.setObjectName("label_38")
        self.label_39 = QtWidgets.QLabel(Form)
        self.label_39.setGeometry(QtCore.QRect(30, 530, 491, 31))
        font = QtGui.QFont()
        font.setPointSize(20)
        font.setBold(True)
        font.setWeight(75)
        self.label_39.setFont(font)
        self.label_39.setAlignment(QtCore.Qt.AlignCenter)
        self.label_39.setObjectName("label_39")
        self.lbPredict = QtWidgets.QLabel(Form)
        self.lbPredict.setGeometry(QtCore.QRect(30, 580, 491, 61))
        palette = QtGui.QPalette()
        brush = QtGui.QBrush(QtGui.QColor(170, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Active,
QtGui.QPalette.WindowText, brush)
        brush = QtGui.QBrush(QtGui.QColor(170, 0, 0))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Inactive,
QtGui.QPalette.WindowText, brush)
        brush = QtGui.QBrush(QtGui.QColor(85, 127, 127))
        brush.setStyle(QtCore.Qt.SolidPattern)
        palette.setBrush(QtGui.QPalette.Disabled,
QtGui.QPalette.WindowText, brush)
        self.lbPredict.setPalette(palette)
        font = QtGui.QFont()
        font.setFamily("OpineHeavy")
        font.setPointSize(37)
        font.setBold(True)
        font.setItalic(False)
        font.setWeight(75)
        self.lbPredict.setFont(font)
        self.lbPredict.setText("")
        self.lbPredict.setAlignment(QtCore.Qt.AlignCenter)
        self.lbPredict.setObjectName("lbPredict")
        self.teB4_2 = QtWidgets.QTextEdit(Form)
```

```
        self.teB4_2.setGeometry(QtCore.QRect(930, 630, 104, 21))
        self.teB4_2.setObjectName("teB4_2")
        self.teR4_2 = QtWidgets.QTextEdit(Form)
        self.teR4_2.setGeometry(QtCore.QRect(560, 630, 104, 21))
        self.teR4_2.setObjectName("teR4_2")
        self.teG4_2 = QtWidgets.QTextEdit(Form)
        self.teG4_2.setGeometry(QtCore.QRect(750, 630, 104, 21))
        self.teG4_2.setObjectName("teG4_2")
        self.label_40 = QtWidgets.QLabel(Form)
        self.label_40.setGeometry(QtCore.QRect(930, 610, 101, 20))
        self.label_40.setAlignment(QtCore.Qt.AlignCenter)
        self.label_40.setObjectName("label_40")
        self.label_41 = QtWidgets.QLabel(Form)
        self.label_41.setGeometry(QtCore.QRect(560, 610, 101, 20))
        self.label_41.setAlignment(QtCore.Qt.AlignCenter)
        self.label_41.setObjectName("label_41")
        self.label_42 = QtWidgets.QLabel(Form)
        self.label_42.setGeometry(QtCore.QRect(750, 610, 101, 20))
        self.label_42.setAlignment(QtCore.Qt.AlignCenter)
        self.label_42.setObjectName("label_42")
        self.label_43 = QtWidgets.QLabel(Form)
        self.label_43.setGeometry(QtCore.QRect(550, 580, 491, 31))
        font = QtGui.QFont()
        font.setPointSize(15)
        font.setBold(True)
        font.setWeight(75)
        self.label_43.setFont(font)
        self.label_43.setAlignment(QtCore.Qt.AlignCenter)
        self.label_43.setObjectName("label_43")
        self.label_44 = QtWidgets.QLabel(Form)
        self.label_44.setGeometry(QtCore.QRect(0, 680, 1061, 31))
        font = QtGui.QFont()
        font.setPointSize(14)
        font.setBold(True)
        font.setWeight(75)
        self.label_44.setFont(font)
        self.label_44.setAlignment(QtCore.Qt.AlignCenter)
        self.label_44.setOpenExternalLinks(True)

self.label_44.setTextInteractionFlags(QtCore.Qt.LinksAccessibleByM
ouse)
        self.label_44.setObjectName("label_44")
        self.label_45 = QtWidgets.QLabel(Form)
        self.label_45.setGeometry(QtCore.QRect(310, 660, 261, 31))
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_45.setFont(font)
        self.label_45.setAlignment(QtCore.Qt.AlignCenter)
        self.label_45.setObjectName("label_45")
        self.comboBox = QtWidgets.QComboBox(Form)
        self.comboBox.setGeometry(QtCore.QRect(210, 110, 211, 22))
        self.comboBox.setObjectName("comboBox")
        self.comboBox.addItem("")
        self.comboBox.addItem("")
        self.comboBox.addItem("")
```

```
        self.label_26 = QtWidgets.QLabel(Form)
        self.label_26.setGeometry(QtCore.QRect(140, 110, 47, 13))
        self.label_26.setObjectName("label_26")

        self.retranslateUi(Form)
        QtCore.QMetaObject.connectSlotsByName(Form)


    def retranslateUi(self, Form):
        _translate = QtCore.QCoreApplication.translate
        Form.setWindowTitle(_translate("Form", "Form"))
        self.label.setText(_translate("Form", "R"))
        self.label_2.setText(_translate("Form", "G"))
        self.label_3.setText(_translate("Form", "B"))
        self.label_4.setText(_translate("Form", "A"))
        self.label_5.setText(_translate("Form", "B"))
        self.label_6.setText(_translate("Form", "A"))
        self.label_7.setText(_translate("Form", "R"))
        self.label_8.setText(_translate("Form", "G"))
        self.label_9.setText(_translate("Form", "G"))
        self.label_10.setText(_translate("Form", "A"))
        self.label_11.setText(_translate("Form", "R"))
        self.label_12.setText(_translate("Form", "B"))
        self.label_13.setText(_translate("Form", "R"))
        self.label_14.setText(_translate("Form", "R"))
        self.label_15.setText(_translate("Form", "B"))
        self.label_16.setText(_translate("Form", "B"))
        self.label_17.setText(_translate("Form", "R"))
        self.label_18.setText(_translate("Form", "G"))
        self.label_19.setText(_translate("Form", "A"))
        self.label_20.setText(_translate("Form", "G"))
        self.label_21.setText(_translate("Form", "A"))
        self.label_22.setText(_translate("Form", "A"))
        self.label_23.setText(_translate("Form", "B"))
        self.label_24.setText(_translate("Form", "G"))
        self.pbStart.setText(_translate("Form", "Start Program"))
        self.pbStop.setText(_translate("Form", "Stop Program"))
        self.label_25.setText(_translate("Form", "Passion Fruit
Auto Shorting Machine"))
        self.pbKalibrasi.setText(_translate("Form",
"Callibration"))
        self.label_32.setText(_translate("Form", "Live Camera"))
        self.label_33.setText(_translate("Form", "Side 1"))
        self.label_34.setText(_translate("Form", "Side 2"))
        self.label_35.setText(_translate("Form", "Side 3"))
        self.label_36.setText(_translate("Form", "Side 4"))
        self.label_37.setText(_translate("Form", "Side 5"))
        self.label_38.setText(_translate("Form", "Side 6"))
        self.label_39.setText(_translate("Form", "PREDICT"))
        self.label_40.setText(_translate("Form", "UNRIPE"))
        self.label_41.setText(_translate("Form", "RIPE"))
        self.label_42.setText(_translate("Form", "NEARLY RIPE"))
        self.label_43.setText(_translate("Form", "Fruit
s"))
        self.label_44.setText(_translate("Form",
<head/><body><p><a
https://www.facebook.com/tenri.sidehabi\"><span style=\"
```

```
text-decoration: underline; color:#0000ff;\">SITTI WATENRIAJENG
SIDEHABI</span></a></p></body></html>"))
        self.label_45.setText(_translate("Form", "design copyright
: "))
        self.comboBox.setItemText(0, _translate("Form", "Matang"))
        self.comboBox.setItemText(1, _translate("Form",
"Mengkal"))
        self.comboBox.setItemText(2, _translate("Form", "Muda"))
        self.label_26.setText(_translate("Form", "Klasifikasi"))
        self.pbStart.clicked.connect(self.Start)
        self.pbStop.clicked.connect(self.Stop)
        self.pbKalibrasi.clicked.connect(self.Kalibrasi)

    def Start(self):
        print("Program Started")
        global cap
        global programSetup
        global programRunning
        global ard
        currentFrame = 0

        while(programSetup):
            #setupdatabase dan SVM Model

            self.file = open("testfile.txt","w")

            dbJenis = []
            dbData = []

            print("Try to Fetching data form Database")
            try:
                c.execute("SELECT jenis, r1, g1, b1, a1, r2, g2,
b2, a2, r3, g3, b3, a3, r4, g4, b4, a4, r5, g5, b5, a5, r6, g6,
b6, a6 FROM data_markisa_backup")
                result = c.fetchall()
                if result is None:
                    print("Data Base Loaded...")
            except:
                print("can't read data form database")

            for dt in result:

dbData.append([dt[1],dt[2],dt[3],dt[4],dt[5],dt[6],dt[7],dt[8],dt[
9],dt[10],dt[11],dt[12],dt[13],dt[14],dt[15],dt[16],dt[17],dt[18],
dt[19],dt[20],dt[21],dt[22],dt[23],dt[24]])
                dbJenis.append(dt[0])

            y_jenis = (np.array(dbJenis).astype(np.float))
            x_data = (np.array(dbData).astype(np.float))

            y_jenis.reshape(-1,1)
            x_data.reshape(-1,1)
            clf = SVC(C=25, gamma = 1e-5, degree = 3, coef0=0.001,
'rbf')
            clf.fit(x_data, y_jenis)
            print("Database Trained Successfully")
```

```
            #setup camera
            cap = cv2.VideoCapture(0)
            if not cap.isOpened():
                print("No Camera Attached")
            #set resolusi dan fps kamera
            cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640)
            cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 360)
            cap.set(cv2.CAP_PROP_FPS, 24)
            currentFrame = 0

            programSetup = False
            ard.flush()
            kirim = True
            t1 = time()
            putar = True
            detek = True
        while (programRunning):

            #live cam started
            ret,frames = cap.read()
            #resize frame agar muat pada graphics view
            frames = cv2.resize(frames,None,fx=0.7, fy=0.7,
interpolation = cv2.INTER_CUBIC)
            #print(frames.shape)
            #resize frame untuk mereduksi jumlah piksel yang akan
di proses
            frame = cv2.resize(frames,None,fx=0.5, fy=0.5,
interpolation = cv2.INTER_CUBIC)
            print(frame.shape)
            #'''
            print(kirim)

            print(putar)

            if(kirim):
                ard.write(str.encode("s"))
                kirim=False

            if(detek):
                msg = ard.readline()

            print(msg)

            if msg == b'r' :
                t0 = time()

                if(putar):
                    ard.write(str.encode("a"))
                    putar = False

                    detek = False
                    #ard.flush()
                    #msg = ""
                if(currentFrame == 7):
                    frame1 = frame[20:250,40:160]
                    #cv2.imshow('frame 1',frame1)
                if(currentFrame == 9):
```

```
                    frame2 = frame[20:250,40:160]
                    #cv2.imshow('frame 2',frame2)
                if(currentFrame == 10):
                    frame3 = frame[20:250,40:160]
                    #cv2.imshow('frame 3',frame3)
                if(currentFrame == 11):
                    frame4 = frame[20:250,40:160]
                    #cv2.imshow('frame 4',frame4)
                if(currentFrame == 12):
                    frame5 = frame[20:250,40:160]
                    #cv2.imshow('frame 5',frame5)
                if(currentFrame == 13):
                    frame6 = frame[20:250,40:160]
                    #cv2.imshow('frame 6',frame6)


                    x1, b1, g1, r1, a1 = kmeans(frame1)
                    self.teR1.setText(str(r1))
                    self.teG1.setText(str(g1))
                    self.teB1.setText(str(b1))
                    self.teA1.setText(str(a1))
                    img = cv2.cvtColor(x1, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi1.setScene(self.scene)

                    print(b1, g1, r1, a1 )
                    x2, b2, g2, r2, a2 = kmeans(frame2)
                    self.teR2.setText(str(r2))
                    self.teG2.setText(str(g2))
                    self.teB2.setText(str(b2))
                    self.teA2.setText(str(a2))
                    img = cv2.cvtColor(x2, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi2.setScene(self.scene)

                    print(b2, g2, r2, a2 )
                    x3, b3, g3, r3, a3 = kmeans(frame3)
                    self.teR3.setText(str(r3))
                    self.teG3.setText(str(g3))
                    self.teB3.setText(str(b3))
                    self.teA3.setText(str(a3))
                    img = cv2.cvtColor(x3, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
```

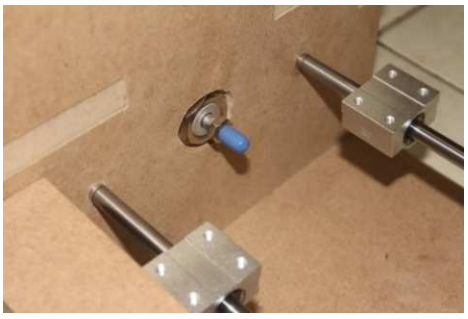
```
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi3.setScene(self.scene)

                    print(b3, g3, r3, a3)
                    x4, b4, g4, r4, a4 = kmeans(frame4)
                    self.teR4.setText(str(r4))
                    self.teG4.setText(str(g4))
                    self.teB4.setText(str(b4))
                    self.teA4.setText(str(a4))
                    img = cv2.cvtColor(x4, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi4.setScene(self.scene)

                    print(b4, g4, r4, a4)
                    x5, b5, g5, r5, a5 = kmeans(frame5)
                    self.teR5.setText(str(r5))
                    self.teG5.setText(str(g5))
                    self.teB5.setText(str(b5))
                    self.teA5.setText(str(a5))
                    img = cv2.cvtColor(x5, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi5.setScene(self.scene)

                    print(b5, g5, r5, a5)
                    x6, b6, g6, r6, a6 = kmeans(frame6)
                    self.teR6.setText(str(r6))
                    self.teG6.setText(str(g6))
                    self.teB6.setText(str(b6))
                    self.teA6.setText(str(a6))
                    img = cv2.cvtColor(x6, cv2.COLOR_BGR2RGB)
                    height, width, channel = img.shape
                    pbl = channel * width
                    imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                    self.scene = QGraphicsScene()
                    self.scene.addPixmap(QPixmap.fromImage(imgF))
                    self.gvSisi6.setScene(self.scene)

                    print(b6, g6, r6, a6)
```

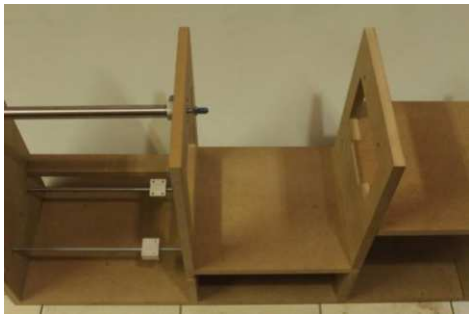
```
                    tkmeans=time()


                    if (self.comboBox.currentText() == "Matang"):
                        dbJenis = 1
                    elif (self.comboBox.currentText() ==
"Mengkal"):
                        dbJenis = 2
                    else:
                        dbJenis = 3
                    dbr1 = r1; dbg1 = g1; dbb1 = b1; dba1 = a1
                    dbr2 = r2; dbg2 = g2; dbb2 = b2; dba2 = a2
                    dbr3 = r3; dbg3 = g3; dbb3 = b3; dba3 = a3
                    dbr4 = r4; dbg4 = g4; dbb4 = b4; dba4 = a4
                    dbr5 = r5; dbg5 = g5; dbb5 = b5; dba5 = a5
                    dbr6 = r6; dbg6 = g6; dbb6 = b6; dba6 = a6
                    #print(dbJenis)

#print(dbJenis,dbr1,dbb1,dbg1,dba1,dbr2,dbb2,dbg2,dba2,dbr3,dbb3,d
bg3,dba3,dbr4,dbb4,dbg4,dba4,dbg5,dbb5,dbg5,dba5,dbr6,dbb6,dbg6,db
a6)

                    sql =  "INSERT INTO data_markisa_backup2
(jenis, r1, g1, b1, a1, r2, g2, b2, a2, r3, g3, b3, a3, r4, g4,
b4, a4, r5, g5, b5, a5, r6, g6, b6, a6 ) VALUES (%s, %s, %s, %s,
%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s,
%s, %s, %s, %s, %s)"

                    try:
                        #c.execute(sql,( dbJenis, r1, g1, b1, a1,
r2, g2, b2, a2, r3, g3, b3, a3, r4, g4, b4, a4, r5, g5, b5, a5,
r6, g6, b6, a6))
                        c.execute(sql,( str(dbJenis), str(dbr1),
str(dbg1), str(dbb1), str(dba1), str(dbr2), str(dbg2), str(dbb2),
str(dba2), str(dbr3), str(dbg3), str(dbb3), str(dba3), str(dbr4),
str(dbg4), str(dbb4), str(dba4), str(dbr5), str(dbg5), str(dbb5),
str(dba5), str(dbr6), str(dbg6), str(dbb6), str(dba6)))
                        db.commit()
                        print("Data Berhasil di Upload")
                    except ValueError:
                        print(ValueError)

                    except:
                        db.rollback()
                        print("Failed to Insert data to Database")
                    #db.close()


                    test = []
                    test.append([str(r1),str(g1),str(b1),str(a1),
                                 str(r2),str(g2),str(b2),str(a2),
                                 str(r3),str(g3),str(b3),str(a3),
                                 str(r4),str(g4),str(b4),str(a4),
                                 str(r5),str(g5),str(b5),str(a5),
str(g6),str(b6),str(a6),])
                    test_data = (np.array(test).astype(np.float))
```

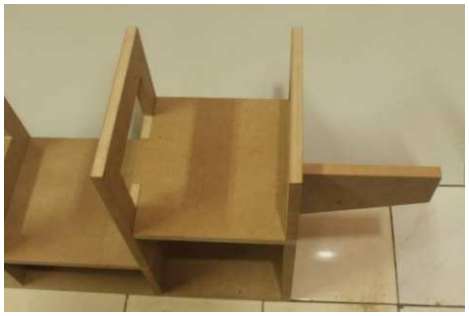
```
                        predict = clf.predict(test_data)
                        if (predict == [1.]):
                            hasil = "RIPE"
                            ard.write(str.encode("1"))
                        elif (predict == [2.]):
                            hasil = "NEARLY RIPE"
                            ard.write(str.encode("2"))
                        elif (predict == [3.]):
                            hasil = "UNRIPE"
                            ard.write(str.encode("3"))
                        self.lbPredict.setText(str(hasil))
                        self.file.write("%s \t"%hasil)
                        self.file.write("waktu eksekusi kmeans %f
\t"%(tkmeans-t0))
                        tfull = time()
                        self.file.write("waktu eksekusi per buah %f
\n"%(tfull-t1))
                        #tm.sleep(3.5)
                        kirim = True
                        putar = True
                        detek = True
                        ard.flush()
                        t1 = time()
                        currentFrame=0

                    currentFrame += 1

                else:
                    currentFrame = 0


                if cv2.waitKey(1) & 0xFF == ord('q'):
                    programRunning = False
                    break


                img = cv2.cvtColor(frames, cv2.COLOR_BGR2RGB)
                height, width, channel = img.shape
                pbl = channel * width
                imgF = QImage(img.data, width, height, pbl,
QtGui.QImage.Format_RGB888)

                self.scene = QGraphicsScene()
                self.scene.addPixmap(QPixmap.fromImage(imgF))
                self.gvLiveCam.setScene(self.scene)



    def Stop(self):
        print("Program Stopped")
        global cap
        cap.release()
        #cv2.destroyAllWindows()
        self.live = False
        programRunning = False
        print("Cleaning...")
```

```
            #GPIO.cleanup()
            print("Byee")
            #sys.exit()
            self.file.close()

        def Kalibrasi(self):
            print("Calibration Machine")
            ard.write(str.encode("s"))

            msg= ard.readline()
            print(msg)

            if msg == b'':
                ard.write(str.encode("a"))
                tm.sleep(3)
                ard.write(str.encode("1"))
            kirim = True

            putar = True
            detek = True

if __name__ == '__main__':

    try:
        db =
MySQLdb.connect("localhost","root","50er412di","db_markisa")
        c = db.cursor()
        print ("Server Connected....")
    except:
        print ("Server Offline....")

    app = QtWidgets.QApplication(sys.argv)
    ex = Ui_Form()
    ex.show()
    sys.exit(app.exec_())
```

E. PROSES PERAKITAN MESIN PEMILAH BUAH MARKISA







F. SURVEI KE INDUSTRI SIRUP MARKISA AURORA

G. SURVEI KE PETANI PENGHASIL BUAH MARKISA DI JENEPONTO