CHAPTER 06

CONCLUSIONS

This software model is developed as a location specified model, to the coordinates of University of Moratuwa (6.7969° N, 79.9018° E). The prediction results can be used real time for a solar farm at the location to maintain and improve the power system balance and stability, whether it is a standalone system or a grid connected farm.

6.1 Model Limitations

The prediction of solar irradiance is based on the output imagery of the developed hardware by university students. The performance of the hardware are not up to the extreme end performance like commercial total sky imagers. The coverage and image quality variations effect the accuracy of results.

The obtained results of the image processing model are approximately accurate for 10 seconds of duration. As a short term prediction model, these results can be used in immediate system control decisions. The temporal resolution can be improved with higher quality images and further developed prediction algorithm.

6.2 Future Research and Applications

The hardware can be developed in to a fully automated system with a mini computer installed on board to output the prediction results real-time. Also the processed data can be transferred to a main system or a data logging system through a wireless communication module for further analysis as an improvement.

The model can be improved with integration of ground specific variables like temperature, wind speed which affects the output of a solar panel. Additionally, software constants can be separately fine-tuned for different months based on the monsoon effect for the location on each month.

Also the integrated hardware and software module can be installed in other locations and modify the prediction algorithm for those locations. To apply the developed algorithm in to a different location, the constants of the model should be fine-tuned using collected data of the location.

Also if a different type and size of a solar panel is used instead of 5W polycrystalline panel, the variable I_{max} will be a different value. Hence the model has to be adjusted by fine-tuning the constants with location specific solar irradiance data. The solar panel performance is assumed to be constant throughout the considered time span.

A solar prediction model running with a hardware is very useful in integrating a solar farm to a grid system or using standalone. This developed model can be used to predict the energy harvest short-term, where electricity generation and consumption can be mapped real-time.

REFERENCES

[1] V.T.Jayadevan, V.P.A. Lonij, "Forecasting solar power intermittency using ground based cloud imaging", *Proceedings of American Solar Energy Society Meeting* (2012)

[2] V. Kostylev, A. Pavlovski, "Solar power forecasting performance - towards industry standards", *Workshop on the integration of solar, 2011*

[3] C.Cornaro, F Bucci, M. Pierro, F.Del Frate, S. Peronaci, A. Taravat, "solar radiation forecast using neural networks for the prediction of grid connected pv plants energy production (dsp project)", *Weather Intelligence for Renewable Energies conference*

[4] S.Sendanayake, M.P.Miguntanna, M.T.R.Jayasinghe, " Estimating incident solar radiation in tropical islands with short term weather data" , *European Scientific Journal January 2014*

[5] C. N. Long, D. W. Slater T. Tooman, "Total Sky Imager Model 880 Status and Testing Results", November 2001

[6] T. Hashimoto, Y. Nagakura, "Prediction of output power variation of solar power plant by image measurement of cloud movement", *Journal of Advanced Research in Physics 2(2), 021107 (2011)*

[7] Long, C.N., Sabburg, J.M., Calbé, J., Pagès, D., 2006. Retrieving cloud characteristics from ground - based day time color all - sky images . J.Atmos.OceanicTechnol.23,633–652)

[8] A. Kazanzidis, P. Tsoumanikas,A.F.Bais,F.Photopaulos,G. Economou, "Cloud detection and classification with the whole sky ground based images", *Atmospheric research 2012*

[9] Yankee Enviromental Systems,inc. (2006) TSI-880 automatic total sky imager,[Online]. Available: http://www.vesinc.com/products/data/tsi880/.

[10] A.S. Rodrigo, K.H.E.Perera, H.M.S. Priyadharshana, V.G.C. Priyanka, R.A.R.A. Ranasinghe, "Cloud images capturing system for solar power level prediction," *Moratuwa Engineering Research Conference (MERCon), April 7-8, 2015.*

APPENDIX A

MATLAB code of the software model

```
clear
im1=imread('output2\frame-150.jpg');%read first image
imd=double(im1);
imr=imd(:,:,1);
imb=imd(:,:,2);
imdiff=imb-1.02*imr;% this const to be fine tuned
imh=histeq(imdiff);
L=graythresh(imh);
imbw=im2bw(imh,L); % convert image in to B&W
imc=imcomplement(imbw);

seo1 = strel('disk',10);
isp1=imopen(imc,seo1);

label1=bwlabel(isp1);
R = regionprops(isp1,'Area'); % preparing area filter
ind = find([R.Area] >= 500);

filteredIm1 = ismember(label1,ind); %applying area filter
processedIm1 = bwlabel(filteredIm1);
%identify sun shade1, give centroid limits
im1labels = max(max(processedIm1))
%get max of columns and get max of them
stats1 =
regionprops(logical(filteredIm1),'Centroid','Area','MajorAxisL
ength','MinorAxisLength','EquivDiameter');
sunshade1=[0,0,0,0];%Centroid(1),Centroid(2),Area,label
for k = 1:im1labels
    data = stats1(k)
    if(abs(data.Area-6700)<3000)%area limit for initial
sunshade
        if(abs(1-
(data.MajorAxisLength/data.MinorAxisLength))<0.5)
            if((abs((data.Centroid(1)-900))<5) &&
(abs((data.Centroid(2)-500))<20))%decided from centroid data
of given video
                if(sunshade1==[0,0,0,0])%if no sunshade1 so
far set this

sunshade1=[data.Centroid(1),data.Centroid(2),data.Area,k];
                elseif(abs(data.Area-6700)< abs(sunshade1(3)-
6700))%check which data nearest to 6700

sunshade1=[data.Centroid(1),data.Centroid(2),data.Area,k];
                end
            end
        end
    end
end
index = 1;
Frame_No = 150;
Reference_array(1,:) = 1:100;% an array to keep track of the
same cloud while labels are changing, 2nd row filled with
im1labels
%define the no. of columns with a column index
Reference_array(2:62,1) = 0:60;%filling index column up to
1hour
Reference_array(2,2:im1labels+1)= 1:im1labels;%2nd row up to
im1 labels
Reference_array(2,2) = sunshade1(4);%writing 1st sunshade
label at row2
Reference_array(2,sunshade1(4)+1) = 1;%giving a place to
replaced label no.01
last_column_index = im1labels+1;%keep track of the last
column, +1
missed_clouds = 0;
pre_missed_clouds = 0;
total_missed_clouds = 0;
%---------------------------------------------------------------
-
while(index<300)% no of frames to go
    if(index>1) %if it is not the first round
      im1 = im2;
      im1labels = im2labels;
      stats1 = stats2;
      sunshade1 = sunshade2;
    end
Frame_No = Frame_No + 450;%frame after 30 sec
Address = strcat('output2\frame-',int2str(Frame_No),'.jpg');
im2=imread(Address);% read next image
%im2 processing
imd=double(im2);
imr=imd(:,:,1);
imb=imd(:,:,2);
imdiff=imb-1.02*imr;
imh=histeq(imdiff);% histogram equalization
L=graythresh(imh);
imbw=im2bw(imh,L); % convert image in to B&W
imc=imcomplement(imbw);

seo1 = strel('disk',10);
isp1=imopen(imc,seo1);

label1=bwlabel(isp1);
R = regionprops(isp1,'Area'); % preparing area filter
ind = find([R.Area] >= 500);

filteredIm2 = ismember(label1,ind); %applying area filter
processedIm2 = bwlabel(filteredIm2);
```

```
%imshow(filteredIm2);
im2labels = max(max(processedIm2))%get max of columns and get
max of them
stats2 =
regionprops(logical(filteredIm2),'Centroid','Area','MajorAxisL
ength','MinorAxisLength','EquivDiameter');
%above labeled matrix converted in to a logical matrix to save
memory
%identify sun shade2
sunshade2=[0,0,0,0];%Centroid(1),Centroid(2),Area,label
cloud_factor = sum(imc(:))/(1280*720-6700);
range1 =
strcat('A',int2str(uint16(index*5)),':A',int2str(uint16(index*
5)));
xlswrite('Prediction.xlsx',cloud_factor,2,range1);
for m = 1:im2labels % to find sunshade2
    data = stats2(m);
    if(abs(sunshade1(1)-data.Centroid(1))<50) &&
(abs(sunshade1(2)-data.Centroid(2))<50)
        if(abs(data.Area-6700)<3000)
         if(sunshade2==[0,0,0,0])
sunshade2=[data.Centroid(1),data.Centroid(2),data.Area,m];
            elseif(abs(data.Area-6700)< abs(sunshade2(3)-
6700))%compare area to find best match

sunshade2=[data.Centroid(1),data.Centroid(2),data.Area,m];
         end
       end
    end
end
if(index==1)
    Reference_array(3,sunshade1(4)) = sunshade2(4);%selecting
reference column for sunshade
    Reference_array(index+2,2) = sunshade2(4);%writing
sunshade value2 in reference array column2
else  for row_two_value = 1:last_column_index% if 26 columns
exist
        if(Reference_array(2,row_two_value)== sunshade1(4))
            Reference_array(3,row_two_value)= sunshade2(4);
            break;
        end
      end
end
 matched_array = true([1 im2labels]);%defininining a logic array
to filter selected clouds
matched_array(uint8(sunshade2(4)))=0;%skipping 2nd sunshade
from matching algo
matched_cloud = [0,0,0,0,0];% define an array for matched
centroid, Centroid(1),Centroid(2),Area,label,eqD
%maxrow3column = 0;
  for k = 1:im1labels %image1 index
```

```
    if (k == sunshade1(4)) %skipping sunshade1 from searching
algo
        continue;
    end
    a=matched_cloud(4);%a gets a value if a cloud from im2 is
matched to previous k
    if((k~=1) && (a~=0))
        matched_array(uint8(a))=0;
    end %skiping selected clouds, k not required, not
 applicable at first round
    matched_cloud = [0,0,0,0,0];% resetting array for matched
centroid, Centroid(1),Centroid(2),Area,label,eqD
    velocity = 0;
    coverX = 0;
    coverY = 0;
    time_to_cover_sun = 0;
    covering_duration = 0;
    x = stats1(k).Centroid;%image1 data
for m = 1:im2labels % to match a m for the current k
 if(matched_array(m) == 1)%if no cloud is matched for m so far
        y = stats2(m).Centroid;%image2 data
        if (abs((y(1)-x(1)))<100 && abs((y(2)-x(2)))<100)%
check centroid values
            if(abs(stats2(m).Area-stats1(k).Area) <
2700)%check area
                if(abs(stats2(m).MajorAxisLength-
stats1(k).MajorAxisLength) < 500)%check shape
                    if(matched_cloud == [0,0,0,0,0])
                        matched_cloud =
[y(1),y(2),stats2(m).Area,m,stats2(m).EquivDiameter];
                    elseif (abs(stats2(m).Area-
stats1(k).Area)<(abs(matched_cloud(3)-stats1(k).Area)))%select
lowest area difference
                        matched_cloud =
[y(1),y(2),stats2(m).Area,m,stats2(m).EquivDiameter];


                    end
                end
            end
        end
    end
end
for n = 2:last_column_index % to write matched labels to ref.
array
        if (Reference_array(index+1,n)==k)% finding ref.column
related to current k label
            Reference_array(index+2,n) = matched_cloud(4);
            if maxrow3column < row2column
                maxrow3column = row2column;
            end
            break;
        end
        end
      end
        % count total missed clouds
        for n = 2:last_column_index
            if (Reference_array(index+2,n) == 0)
              total_missed_clouds = total_missed_clouds+1;
            end
        end
        missed_clouds = total_missed_clouds -
pre_missed_clouds;%calc missed_clouds for this frame
  pre_missed_clouds = total_missed_clouds;%updating pre missed
clouds

        matched_array = true([1 im2labels]);%redefining the
logic array to filter missed clouds
for n = 1:im2labels % to find not matching labels// this
loop's result is same as current value
           for m = 2:last_column_index % searching through all the
used columns
                if(Reference_array(index+2,m) == n)
                    matched_array(1,n) = 0; %making matched labels
zero
                end
            end
end
    %writing missed_cloud values to ref array
    for x = 1:im2labels
        if(matched_array(1,x) == 1)   %selecting missed values
          Reference_array(index+2,last_column_index+1) = x;
%writing missed values
          last_column_index = last_column_index + 1;%increment
last_column_index
        end
    end
%     Reference_array
% eval(sprintf('Prediction_array_%i =
zeros(62,last_column_index*2);',index));
    Prediction_array_index = zeros(62,last_column_index*2+1);
Prediction_array_index(1:62,1) = 0:61;%index column

    %calc velocity
    for x = 2: last_column_index %column range in ref array
        if(Reference_array(index+1,x) ~= 0 &&
Reference_array(index+2,x) ~= 0)%there exists a matched cloud
in lower row
            %xlswrite('ref_array.xlsx', Reference_array);%to
check ref array values
            %stats1(Reference_array(index+1,x));
          Centroid1X =
stats1(Reference_array(index+1,x)).Centroid(1);
          Centroid1Y =
stats1(Reference_array(index+1,x)).Centroid(2);
```

```
%            EqDiameter1 =
stats1(Reference_array(3,x)).EquivDiameter;
          Centroid2X =
stats2(Reference_array(index+2,x)).Centroid(1);
          Centroid2Y =
stats2(Reference_array(index+2,x)).Centroid(2);
          EqDiameter2 =
stats2(Reference_array(index+2,x)).EquivDiameter;

         Vx = (Centroid2X-Centroid1X)/0.5% if +ve moving right
(v per min)
         Vy = (Centroid2Y-Centroid1Y)/0.5% if +ve moving down

         % predict next cloud positions each minute
         Prediction_array_index(1,2*(x-1))= Centroid1X; % if x
=2, these are sunshade values
         Prediction_array_index(1,2*x-1)= Centroid1Y;%given x
range starts from 2
         Prediction_array_index(2,2*(x-1))=
Centroid2X;%therefore x replaced with (x-1)at x centroid
column logic
         Prediction_array_index(2,2*x-1)= Centroid2Y;

            for minute = 3:60
                Prediction_array_index(minute,2*(x-1)) =
Centroid2X + Vx; %x value to array
                Centroid2X = Centroid2X + Vx;%updating
centroid variable
                Prediction_array_index(minute,2*x-1) =
Centroid2Y + Vy; %y
                Centroid2Y = Centroid2Y + Vy;
            end
         % predict solar coverage
if(Vx>0 && (sunshade2(1)>(Centroid2X)-EqDiameter2/2))% (x -
cloud radius)
                coverX = 1;
            elseif (Vx<0 &&
(sunshade2(1)<(stats2(Reference_array(index+2,x)).Centroid(1)+
stats2(Reference_array(index+2,x)).EquivDiameter/2)))
                coverX = 1;
            end
            if(Vy>0 &&
(sunshade2(2)>(stats2(Reference_array(index+2,x)).Centroid(2)-
stats2(Reference_array(index+2,x)).EquivDiameter/2)))
                coverY = 1;
            elseif (Vy<0 &&
(sunshade2(2)<(stats2(Reference_array(index+2,x)).Centroid(2)+
stats2(Reference_array(index+2,x)).EquivDiameter/2)))
                coverY = 1;
            end

        end
```

```
    end
     Vx = (matched_cloud(1)-x(1))/5;% if -ve moving right
    Vy = (matched_cloud(2)-x(2))/5;% if -ve moving down
    predict solar coverage
    if(Vx>0 && (sunshade2(1)>(matched_cloud(1)-
matched_cloud(4)/2)))% (x -cloud radius)
        coverX = 1;
    elseif (Vx<0 &&
(sunshade2(1)<(matched_cloud(1)+matched_cloud(4)/2)))
        coverX = 1;
    end
    if(Vy>0 && (sunshade2(2)>(matched_cloud(2)-
matched_cloud(4)/2)))
        coverY = 1;
    elseif (Vy<0 &&
(sunshade2(2)<(matched_cloud(2)+matched_cloud(4)/2)))
        coverY = 1;
    end
    calc covering time & duration
    if((coverX == 1) && (coverY == 1))
        Tx = abs((sunshade1(1)-matched_cloud(1)-
(matched_cloud(4)/2))/Vx);%sun considered as a point compared
with cloud size
        Ty = abs((sunshade1(2)-matched_cloud(2)-
(matched_cloud(4)/2))/Vy);
        Dx =
abs((stats2(matched_cloud(4)).EquivDiameter+92.362)/Vx);
%pi*r^2 = 6700
        Dy =
abs((stats2(matched_cloud(4)).EquivDiameter+92.362)/Vy);
        % checking overlap
        if(Tx<Ty)% finding which axis covers first
            R1 = Ty;
        else R1 = Tx;
        end
        if((Tx+Dx)<(Ty+Dy))% finding which axis stops covering
first
            R2 = Tx+Dx;
        else R2 = Ty+Dy;
        end
        if(R1<R2)
            R1 = R1 + (index + 1)*5;%considering 5s rounds,
image 1 is out of this index, so +1
            R2 = R2 + (index + 1)*5;% remove +1 its wrong

        %predict covering percentage
        VT_alpha = Vy/Vx;
        PT_alpha = (sunshade2(2)-
matched_cloud(2))/(sunshade2(1)-matched_cloud(1));
        Delta_alpha = abs(atand(PT_alpha)-atand(VT_alpha));
        covering_percentage = 0;
            if(matched_cloud(3)>6700 && Delta_alpha<7 )
```

```
                covering_percentage = 1;
            elseif(matched_cloud(3)<6700 && Delta_alpha<7 )
                covering_percentage =matched_cloud(3) /6700;
            elseif( Delta_alpha>7 )
                if(matched_cloud(5)/2+46.181-
tan(Delta_alpha)*((sunshade2(1)-
matched_cloud(1))^2+(sunshade2(2)-matched_cloud(2))^2)^0.5<0)
                covering_percentage = 0;
                else covering_percentage
=((matched_cloud(5)/2+46.181-tan(Delta_alpha)*((sunshade2(1)-
matched_cloud(1))^2+(sunshade2(2)-
matched_cloud(2))^2)^0.5))/(2*46.181);
                end
elseif(matched_cloud(3)<6700 && Delta_alpha>7 )
                covering_percentage =0.3;
            else covering_percentage = 0.5;
            end
            write to excel sheet
            range =
strcat('F',int2str(uint16(R1)),':F',int2str(uint16(R2)));
            [rows,columns] = size(Prediction_array_index);
            if columns>256
                error('column limit exceeded');
            end

            range =
strcat('A1',':',char2str(lastcolumn),':',int2str(count));

xlswrite('Prediction.xlsx',covering_percentage,range)
              xlswrite('cloud_prediction.xlsx',
Reference_array,index);%excelsheet, array name, sheet no.
                    range = strcat('A',int2str(index));
              xlswrite('cloud_prediction_final.xlsx',
Prediction_array_index,1,range);
              Prediction_array_index;
       end
     end
end

index = index + 1;
  %Reference_array;
end
%subplot(2,1,1);imshow(processedIm1);title('1');
%subplot(2,1,2);imshow(processedIm2
```