BNL-108159-2015-JA

3D Cloud Detection and Tracking System for Solar Forecast Using Multiple Sky Imagers

Zhenzhou Peng, et al.

Computational Science Center

Brookhaven National Laboratory

U.S. Department of Energy

Advanced Scientific Computing Research (ASCR)

Notice: This manuscript has been authored by employees of Brookhaven Science Associates, LLC under Contract No. DE- SC0012704 with the U.S. Department of Energy. The publisher by accepting the manuscript for publication acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or any third party's use or the results of such use of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof or its contractors or subcontractors. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

BNL-108159-2015-JA

3D Cloud Detection and Tracking System for Solar Forecast Using Multiple Sky Imagers

Zhenzhou Peng[a,b], Dantong Yu[a,b,*], Dong Huang[b,*], John Heiser[b], Shinjae Yoo[b], Paul Kalb[b]

[a] *Department of Electrical and Computer Engineering, Stony Brook University, 100 Nicolls Road, Stony Brook, NY 11790, USA*
[b] *Brookhaven National Laboratory, 2 Center Street, Upton, NY 11973, USA*

Abstract

We propose a system for forecasting short-term solar irradiance based on multiple total sky imagers (TSIs). The system utilizes a novel method of identifying and tracking clouds in three-dimensional space and an innovative pipeline for forecasting surface solar irradiance based on the image features of clouds. First, we develop a supervised classifier to detect clouds at the pixel level and output cloud mask. In the next step, we design intelligent algorithms to estimate the block-wise base height and motion of each cloud layer based on images from multiple TSIs. This information is then applied to stitch images together into larger views, which are then used for solar forecasting. We examine the system's ability to track clouds under various cloud conditions and investigate different irradiance forecast models at various sites. We confirm that this system can 1) robustly detect clouds and track layers, and 2) extract the significant global and local features for obtaining stable irradiance forecasts with short forecast horizons from the obtained images. Finally, we vet our forecasting system at the 32-megawatt Long Island Solar Farm (LISF). Compared with the persistent model, our system achieves at least a 26% improvement for all irradiance forecasts between one and fifteen minutes.

*Corresponding authors
Email addresses: zhenzhou.peng@stonybrook.edu (Zhenzhou Peng), dtyu@bnl.gov (Dantong Yu), dhuang@bnl.gov (Dong Huang), heiser@bnl.gov (John Heiser), sjyoo@bnl.gov (Shinjae Yoo), kalb@bnl.gov (Paul Kalb)

Keywords: sky imagery, cloud detecting, cloud tracking, short-term forecast

1. Introduction

Clouds are the primary source of large fluctuations in solar radiation at commercial solar farms. The main interaction between clouds and radiation in the visible spectrum is scattering, which can rapidly change the gross radiation measured by surface radiation sensors. Cloud-induced variability in solar radiation has become one of the greatest concerns in the power grid, as the market share of solar energy, that is, solar energy penetration, has steadily increased in recent years. Consequently, the ability to predict the presence of clouds and extract their relevant features is critical for estimating variations in solar energy and thus for mitigating the effects of output fluctuations in utility-scale PV power plants.

Methods for detecting and tracking clouds are widely applied to estimate their motion n satellite images at the global scale, viz., spatial scales of 1x1km or larger and temporal scales of 15 minutes or longer, [1] [2] [3]. These methods are not designed for accurately estimating cloud motions on shorter time scales or at smaller spatial scales. In particular, for such applications as distributed residential roof-top solar panels and utility-scale solar farms, cloud tracking requires real-time or near real-time cloud information at a high resolution of meters and minutes, given the enormous volatility of the weather and atmospheric conditions that prevail at this scale. Therefore, instead of a satellite imaging system, localized cloud tracking requires instruments and new sophisticated algorithms to operate at a speed commensurate with that of the rapid changes in clouds. Cloud tracking at such a scale is challenging because they exhibit properties, such as Cloud base heights (CBH) that range from 0 to 7 km, movements in arbitrary directions and at different velocities, ambiguous, non-rigid shapes depending on their types.

To accurately determine the properties of local clouds, it is common to utilize measurements from ground-based instruments, such as cloud radar and

LIDAR. However, because of their high cost and limited availability, these tools are not widely used for localized, short-term forecasts. Although cloud tracking is undertaken based on ground-based reference sensor network [4], additional hardware support is needed for tracking over a large region. A cost-effective alternative is to use sky imager, such as total sky imager (TSI) to infer the clouds' irradiance properties and estimate their movements in a short time window. With the advent of inexpensive digital cameras, sky imagers have become widely used as cost-effective tracking device for various applications.

Compared with satellite imaging system and ground-based reference sensors, sky imager is a promising tool for visualizing and tracking clouds in real time at local scales. Such sky images have been successfully used to estimate observed cloud cover over a short-time horizon [5] [6] [7]. By analyzing the red-green-blue (RGB) channels in sky images, researchers can determine the details of various cloud properties, such as opaqueness and thickness [8] [9] [10]. In addition to utilizing original RGB channels, Souza-Echer et al. proposed detecting clouds in the hue-saturation-luminance (HSL) color space [11]. In our earlier work, we have extracted generalized features from transformed luminance images [12]. Other approaches used such classification methods as state-of-the-art k-nearest neighbor(kNN) algorithms and binary decision tree to categorize cloud types based on the extracted textural features from sky imagery [13] [14]. Recent works [15] [16] [9] [17] have focused on exploring additional image characteristics or the application of adaptive thresholds to improve the accuracy of classifying cloud types and identifying cloud pixels. In particular the reported accuracy of seven-type cloud classification has ranged from 78% to 95% [17].

Sky imagery is also an effective tool for tracking cloud movements over short time horizon which can not be accessed using satellite imaging system. Based on the temporal correlations between consecutive frames of sky images, cloud movement can be obtained as motion vectors at the pixel level. One common approach adopted by previous researchers is to divide sky images into small regions/blocks to obtain block-wise motion vectors. Researchers at the University of California San Diego (UCSD) initially utilized the normalized cross-

correlation method (CCM) to detect the velocities of regional cloud motion in sky images [18]. To improve the accuracy of cloud motion tracking based on small blocks, Huang et al. proposed the adoption of two different block sizes for block matching [19] [20]. To mitigate cloud deformation and motion perturbation, Huang et al. [21] explored an approach based on multi-frame motion back-tracking. The cloud motion vector was then estimated via extrapolation based on the known motion trend. In contrast to block-wise movement, the motions of clouds can also be represented on a tiny scale, e.g., by the motions of individual pixels. Pixel-wise motion is generally estimated by utilizing well-known computer vision techniques. Optical flow (OF) motion tracking, which has the advantages of being able to identify tiny differences between pixels, is a successful example of estimating the small movements of clouds [22]. By incorporating the deformation model, OF-based methods can capture the dynamic motions of clouds between consecutive frames based on image registration techniques [23]. In general, block-wise cloud tracking, as discussed in previous works [24] [25], is an extension of the pixel-wise methodology. These two approaches are applied under different tracking conditions: in the block-based approach, the properties within a cloud block are integrated to mitigate noise and misrepresentations of color in the original images, whereas the pixel-wise approach focuses on the typical movements that occur at the pixel or sub-pixel scale.

Recently, sky imagery has been widely used in solar irradiance studies to predict the presence of clouds based on the temporal correlation between frames. The basis of this approach is predicting the cloud properties in and around the path between the radiation sensor and the Sun based on previously observed clouds and then estimating the cloud-induced fluctuations in irradiance based on these properties [18]. By representing irradiance as a function of the features extracted from sky images, solar prediction can then be transformed into a problem of predicting future images and accurately resolving their corresponding cloud features [26] [12]. This methodology, as might be expected, relies on the ability to detect clouds and track their motion. To minimize the influence of erroneous motions, Huang et al. [21] added statistical features to the final

irradiance forecast model. To further improve the accuracy of short-term solar forecasting, recent works have incorporated machine learning techniques into both cloud tracking and irradiance modeling [27] [20] [28].

In general, sky imagery can be used to effectively determine a cloud's presence, coverage and short-term motion. The majority of existing sky imagers are designed to image a hemispheric view of the sky. Their built-in cameras are either oriented upward, for direct image acquisition, or, downward, to capture the reflections from a spherical mirror [29] [6] [30] [31]. However, the effectiveness of the cloud tracking capabilities and detection methods may vary among different types of sky imagers. In particular, the TSI used herein to capture cloud movements over short horizons has several shortcomings that limit its cloud detection ability and cloud tracking effectiveness. One significant disadvantage of the TSI is that it adjusts its imaging settings in response to the overall scene brightness. Therefore, the raw TSI images captured under different lighting conditions lack inter-image consistency and often cannot faithfully represent the real view. Another issue with the TSI is its built-in camera can acquire only low-resolution images of up to 640 x 480. Moreover, because the TSI images are generated by reflections from a dome-shaped mirror, they typically suffer from the presence of ground obstacles within the FOV. More importantly, the shadow band which occludes approximately 14% of the total view [31] and the camera-supporting arm are visible in the raw images, and must to be removed during the preprocessing step. A promising alternative to TSI is an inexpensive high-definition digital camera, which overcomes these issues of image quality. This approach has been used successfully for short-term solar forecasts [30]. To address these challenges in TSI images, we propose employing a novel system to identify the primary cloud layers and extract the effective image features that are relevant to forecasting the surface radiation. Our contributions are summarized as follow:

1. **New Cloud Detection Methodology:** We propose an effective cloud detection pipeline for classifying cloud pixels and utilizing multiple TSIs to correct erroneous TSI images or compensate for abnormal exposure.

2. **Multiple-Source Cloud Tracking:** In contrast to the related work

described in this section, we design a novel tracking algorithm to incorporate the spatial and temporal correlations between clouds among multiple TSIs and time frames.

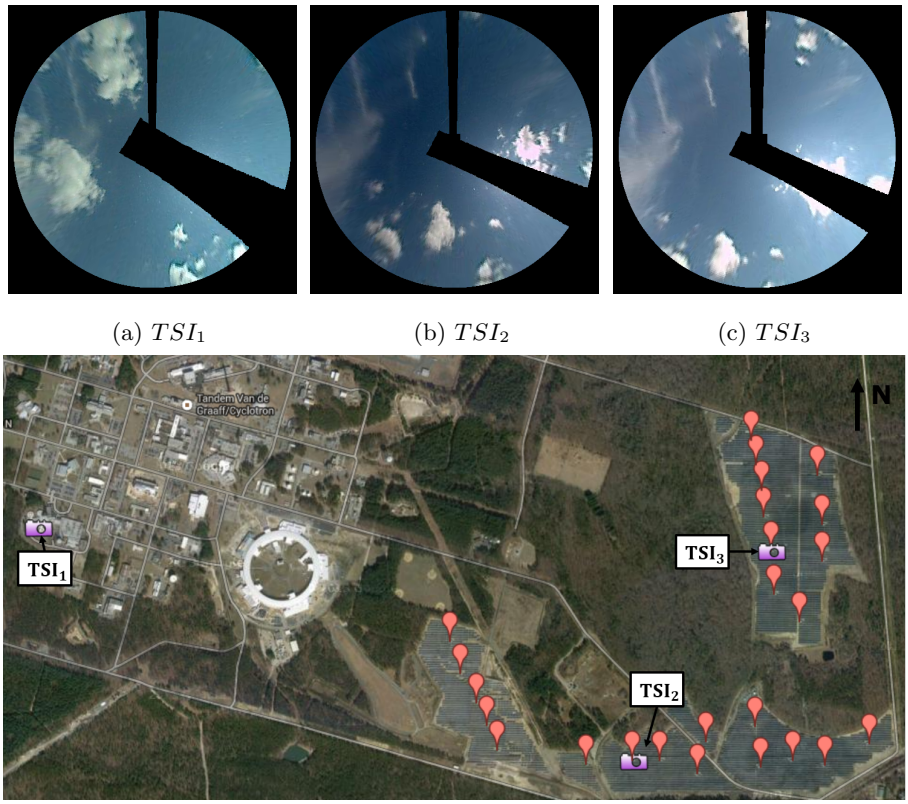
3. **Multi-layer Recognition:** We develop a clustering-based technique to integrate the results of individual block-wise tracking into multiple cloud layers, each of which forms a major wind field.

4. **Feature Selection:** Instead of tracking a single cloud pixel, we use sky images to generate both global features at the image level as well as local variations within a small pixel 7 x 7 window.

5. **New Regression-based Irradiance Forecast Models:** We propose four regression-based solar irradiance models that utilize the predicted features of cloud pixels, and compare these models with the persistent model and its variant based on Support Vector Regression.

6. **High Accuracy in Field Studies and Validation:** We confirm, in our preliminary analysis, that the proposed system for detecting and tracking clouds and predicting their relevant features can accurately forecast solar radiation for up to 15 minutes.

This paper is organized as follows. Section 2 describes our ground-based instruments and data streams. Section 3 presents a pipeline for supervised cloud detection and cloud block aggregation. In Section 4, we provide a detailed description of a 3D cloud block tracking system for detecting multiple cloud layers, each of which is defined as a wind field. The information for each cloud layer then is used to stitch together multiple sky images. Section 5 describes the extraction of the image features based on the combined multi-TSI view, and applications of different regression models to predict solar radiation. The experiments and subsequent analyses are described in Section 6, followed by our conclusions in Section 7.



(a) TSI_1 (b) TSI_2 (c) TSI_3

(d) Google Maps View

Figure 1: (a), (b), and (c) are the preprocessed views of the three TSIs. (d) is the Google Maps view of the three TSIs from left to right (camera iconslabel) and the 25 solar radiation sensors (red). The distance between TSI_1 and TSI_2 is 2477 meters, and that between TSI_2 and TSI_3 is 956 meters.

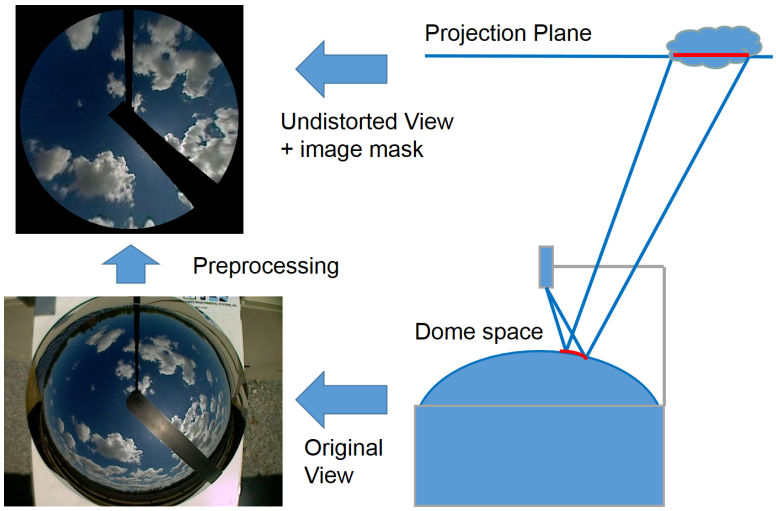


Figure 2: Procedure for preprocessing the TSI images. The original image is undistorted from the original dome space to the planar space via coordinate transformation. The output image is cropped based on a pre-defined FOV range and masked to remove irrelevant areas, such as the supporting arm and the shadow band.

2. Instruments

The data used in this paper were obtained from the Long Island Solar Farm (LISF), a 32-megawatt solar photovoltaic power plant built by a collaboration between BP Solar, the Long Island Power Authority (LIPA), and the Department of Energy. The LISF, located at the Brookhaven National Laboratory, is currently the largest solar photovoltaic power plant in the Eastern United States. The cloud tracking system (Figure 1) consists of a network of three total sky imagers (TSI_1, TSI_2, and TSI_3). Their positions are triangulated to ensure good coverage of the sky above the solar farm. TSI_1 collects sky images of a region near the solar farm. TSI_2 is deployed in the middle of the farm to ensure sufficient overlap of its views with those of the other two TSIs. TSI_3 is located in the northern area of the solar farm. We installed all three TSIs at the same altitude and tuned their orientations to ensure that their camera-supporting arms point north. Consequently, the supporting arm in each TSI images overlaps with the vertical(y) axis on the image which is aligned toward

the north.

All TSIs uploaded real-time video streams with a raw resolution of 640x480, and at a streaming rate of 10 seconds per frame to a centralized database server. We then applied preprocessing techniques, as illustrated in Figure 2, to undistort the raw images and remove unrelated artifacts, i.e., the shadow band and supporting arm. The primary advantage of preprocessing the images, as described in our previous work [19], is that the planar view obtained from the coordinate transformation of the raw images is more effective for estimating cloud motions. In this study, we projected the original coordinates to a flat plane with a given resolution (500x500 pixels) and cropped the planar image using a pre-defined view angle range (zenith range $0^\circ - 60^\circ$) to ensure an FOV of approximately 120°. In addition, because the camera-supporting arm may not be exactly aligned with the vertical direction in a TSI image, we adjusted the orientation in the preprocessing step by rotating the images by several degrees. To simplify the coordinates transformations between different TSIs, we unified the conditions and specifications of all TSI devices. Because each TSI has an identical FOV and resolution, all the preprocessed images reside on the same projection plane, wherein pixels are distributed evenly. Finally, we applied a pre-calculated mask to remove irrelevant areas, such as the supporting arm and shadow band. The preprocessed image was then generated, with all irrelevant pixels being set to black (Figures 1a, 1b, and 1c).

We deployed 25 pyranometers in the LISF, as shown in Figure 1d, to measure the surface solar irradiance. These sensors measure the global horizontal irradiance (GHI) in real time. The measurements which are recorded every 10 seconds, are synchronized with the TSI observation. The variations in zenith and the diurnal and seasonal patterns are also recorded in the raw GHI measurements, and therefore bias our subsequent irradiance forecasting models. To mitigate this potential bias, we normalized each radiation value to a clear-sky index k^t during model training and testing. Letting GHI^t be the raw GHI measured at time t and GHI^t_{clear} be the corresponding clear-sky estimate, the

clear-sky index k^t is calculated as follows:

$$k^t = \frac{GHI^t}{GHI^t_{clear}} \tag{1}$$

where k^t nominally ranges from 0 to 1. However, its maximum value can be greater than one due to the cloud enhancement caused by diffuse sunlight. The estimated clear-sky value GHI^t_{clear} at time t is obtained from the regression curve that best fits the distribution of historical observations [24].

3. Cloud Detection

3.1. Cloud Pixel Identification

The basic concept of cloud detection is to apply a classification methodology to identify cloudy pixels in sky imagery and to separate the clouds from the sky at the pixel level. Beginning with the mask of classified cloud pixels, we aggregate the pixels into appropriate regions/blocks to represent pieces of cloud for subsequent tracking.

However, many challenges are encountered when identifying clouds at the pixel level. With their distinct properties, such as different optical depths and brightness variations in sky imagery, clouds have various distributions in the RGB channels and a wide range of brightnesses in different sky scenes or image frames. In particular, certain optically thin clouds may have similar textures and colors to those of the background sky pixels, whereas some clouds may appear both dark and white within the same sky image. Therefore, it is difficult to identify a fixed threshold to separate clouds from sky. For instance, cirrus clouds are barely distinguishable from the clear-sky background in TSI images.

Moreover, under different weather conditions and at different solar angles, the clouds presented on a bitmap image may appear to have various brightnesses and a large range of intensities. Therefore, we require a robust methodology to assess sophisticated image-based features to capture the pixel-wise differences or the regional textural differences between cloudy and clear regions. From the perspective of image processing, it may appear that we could first sharpen edges

and boundaries using various derivative image filters and then apply image segmentation methods to separate clouds from clear pixels based on their boundary pixels, and thereby identifying cloud segments. However, these methods often fail to segment clouds from the background due to the poorly-defined edges of clouds in sky images. With the resolution and image quality limitations, a single pixel in a sky image may contain both cloud and clear sky. Consequently, a section of visually rigid cloud can appear non-rigid and blurry in sky imagery.

Furthermore, sky imagers face certain instrument-specific challenges in their practical deployment in the field and in testing. Because the sky camera adjusts its lens aperture and shutter speed in response to the amount of incident illumination, the output images can potentially suffer from variations in exposure, and may appear either brighter or darker than the ground-truth image that accurately represents the real lighting conditions.

To accurately detect cloud pixels, previous studies of sky images have relied on cloud properties and have utilized the prominent features at the pixel level to identify hybrid thresholds for the RGB channels or the red-blue ratio (RBR) in cloud images [32] [8]. The threshold-based method presented in [8] is able to accurately identify opaque clouds in images acquired by the Whole Sky Imager (WSI). However, it is less effective for many low-quality commodity cameras, such as TSI and webcam. These cameras lack spectral and neutral filters and adjust their lens apertures for different lighting conditions, which incurs large variations in the brightness of the images and requires non-trivial calibrations in deployment. Under this circumstances, threshold-based methods require customized parameter settings for various sky scenes. Another type of approach to the analysis of cloud properties is based on supervised classification methods that utilize various features extracted from sky images and find patterns from the training dataset. In earlier works, traditional classification techniques, such as k-nearest neighbor (kNN) algorithms, binary decision tree, and neural network classifiers [13] [14] [16] [33], have been widely used for cloud type classification in sky imagery. To detect clouds at the pixel level, a supervised technique can be successfully applied to train the adaptive threshold for

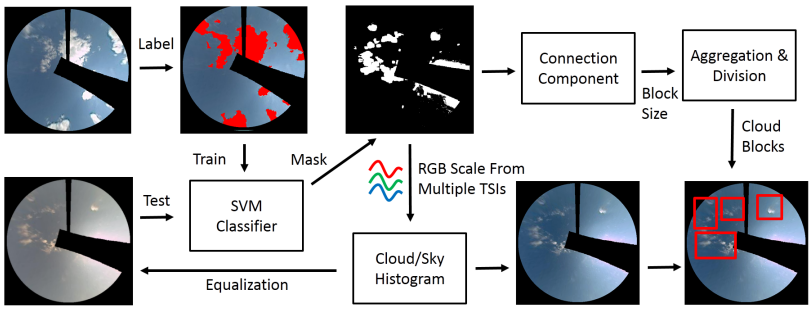


Figure 3: Pipeline for cloud detection using an SVM classifier and multi-source correction.

the classification of image pixels into several opacity categories based on the RBR channel [9]. In this paper, we detail how to use a binary classifier to identify cloud versus clear-sky pixels in sky imagery. The training datasets for the classifier are generated by manually labeling cloud/sky pixels in TSI images. This process requires a considerable amount of human effort and more importantly, may introduce uncertainties and errors into the training datasets that could significantly impact the accuracy of traditional classifiers. To overcome this challenge, we propose an outlier-aware classifier to train manually labeled pixels in sky images. Moreover, because of the many instrument-specific abnormal cases that can be generated from individual TSIs, such as, overexposed or underexposed images, we design a classifier-based pipeline to utilize all three TSIs for multi-source image correction to enhance the overall accuracy of cloud detection. We also present a technique for aggregating cloud pixels into cloud blocks (Figure 3).

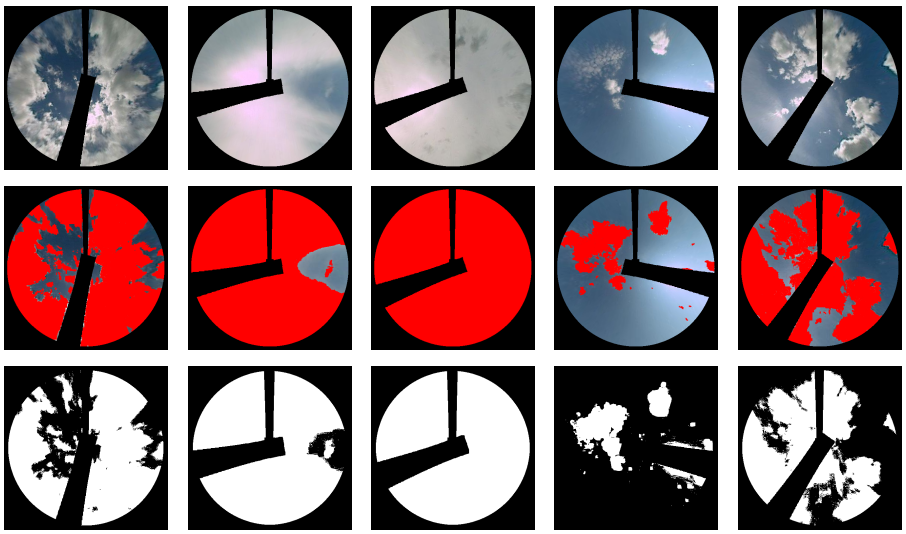
3.2. Supervised Cloud Classifier

To minimize the influence of possible outliers in the training dataset, we chose a Support Vector Machine(SVM) [34] as a pixel-wise classifier and trained it using two-layer cross-validation [35] to reduce overfitting. An SVM constructs a max-margin hyperplane to reduce the effect caused by outliers and offers the advantage of being able to handle known outlier patterns. In particular, uncertainties and errors introduced during the manual annotation of training images

are taken into account during SVM optimization. Figure 3 shows an example of the training and testing process for a cloud classifier. To better describe the characteristics of cloud pixels, six features are extracted from the sky images and normalized for use as the SVM training dataset: R, G, B and RBR which are spectral characteristics extracted from the RGB color space, and two features based on the neighboring pixels which are used to mitigate the impact of variations in illumination. In detail, we convert the sky images from the original RGB color space to the relative luminance space based on the algorithm presented in [36] and then apply the Laplacian of Gaussian(LoG) spatial filter [37] to compute the 2nd derivative in luminance space within a fixed-size window (7x7 is used). The LoG value reflects the rapid changes in the illuminance channel in a certain region and is useful for detecting sharp edges. The final feature used for cloud detection is the standard deviation of the luminance channel within a small region around each pixel (7x7 is used), representing the pixel's average difference from its neighbors. Our implementation uses the SVM package *libsvm* [38] with linear kernel.

Furthermore, we explore the possibility of multi-source abnormality correction using synchronized images from all three TSIs. Because the three TSIs are located reasonably close to each other, we can reasonably assume that their color representations have statistically similar range with regard to the RGB channels. In other words, cloudy/clear pixels in these three digitized color channels should have similar histograms across the different TSIs. Therefore, if one TSI experiences an exposure issue or abnormal brightness in the RGB color space, we can correct it by equalizing the histograms of its RGB channels to those of the two normal TSIs. We first generate the cloud masks for all TSIs using the SVM classifier and compute the histograms of the RGB channels for cloud and sky pixels separately. By calculating the Euclidean distances between the histogram vectors of the three TSIs, we can identify a device as abnormal if its image histogram is significantly different from those of the other two devices. We then apply the histogram equalization add-on to the output image from the abnormal TSI to adjust its RGB scale for cloud and sky pixels. The corrected

result can be used for the next round of cloud mask generation. In practice, we iterate this procedure three times to extract the cloud mask and equalize the RGB histograms of the TSI images (Figure 3).



$AC_{cld} = 93.2\%$	$AC_{cld} = 100\%$	$AC_{cld} = 100\%$	$AC_{cld} = 83.2\%$	$AC_{cld} = 93.8\%$
$AC_{sky} = 87.7\%$	$AC_{sky} = 79.4\%$	$AC_{sky} = 0\%$	$AC_{sky} = 92.9\%$	$AC_{sky} = 85.5\%$

Figure 4: Cloud detection results (row 3) compared to manual annotation (row 2) under different weather/cloud conditions. *Left to Right* in row 1: Scattered cloud, cloudy, overcast, multilayer, and multi-layer with thin cloud. AC_{cld} and AC_{sky} represent the accuracy of the classification results for cloud pixels and sky pixels, respectively.

To evaluate the performance of the proposed supervised classifier and the multi-source correction algorithm, we selected various test cases from daily observations corresponding to different atmospheric conditions and cloud types and then compared the results with our manually annotated images. In this paper, we adopt two evaluation metrics to measure the error in cloud classification:

$$AC_{cld} = \frac{N_{cld,cld}}{N_{cld,cld} + N_{sky,cld}}, AC_{sky} = \frac{N_{sky,sky}}{N_{sky,sky} + N_{cld,sky}} \tag{2}$$

where AC_{cld} and AC_{sky} are the accuracies of cloud and sky pixel classifi-

$AC_{cld} = 97.9\%$	$AC_{cld} = 98.0\%$	$AC_{cld} = 86.1\%$	$AC_{cld} = 98.9\%$	$AC_{cld} = 96.2\%$
$AC_{sky} = 91.5\%$	$AC_{sky} = 84.8\%$	$AC_{sky} = 91.9\%$	$AC_{sky} = 81.9\%$	$AC_{sky} = 91.0\%$

Figure 5: Results of cloud detection (row 3) compared to manual annotation (row 2) in the presence of device-specific bias or luminance variations. *Left to Right* in row 1: red-dominant, green-dominant, overexposed, underexposed, and a different image source.

cation, respectively. $N_{cld,cld}$ and $N_{sky,sky}$ denote the pixel counts of correct cloud and sky classifications, respectively, whereas $N_{sky,cld}$ and $N_{cld,sky}$ indicate the total numbers of sky and cloud pixels, respectively, that are falsely recognized by our detector. In Figure 4, we show the distinctive patterns of examples of scattered clouds, cloudy conditions, overcast conditions, and two multi-layer cases. Compared with the manual classification masks, our pipeline based on the SVM classifier can accurately detect clouds (with an accuracy of more than 83.2%), except in the case of multi-layered clouds near the sun's position and very thin clouds. The image area near the Sun ("sunspot") has a higher brightness and is difficult to characterize based exclusively on static textural information. Therefore, the classifier often falsely labels clear-sky pixels as clouds in that region.

To validate the cloud detection performance of our method in the case of device errors or variations in exposure, we selected four abnormal images of types that are commonly observed in the field: red-color dominant, green-color dominant, overexposed, and underexposed (the first four cases in Figure 5). The accuracy of cloud pixel classification in these cases is 86.1% or higher. We also applied our classifier to another type of sky imagery configured with a different field of view and color scales (the last case in Figure 5), which confirmed that our classification algorithm is practical and effective. The classification accuracies for cloud and sky pixels in this case are 96.2% and 91.0%, respectively. The overall clout detection performance in all selected cases is evaluated in the confusion matrix presented in Table 1. We observed our pipeline accurately (96.6%) recognizes cloud pixels.

3.3. Cloud-block Generation

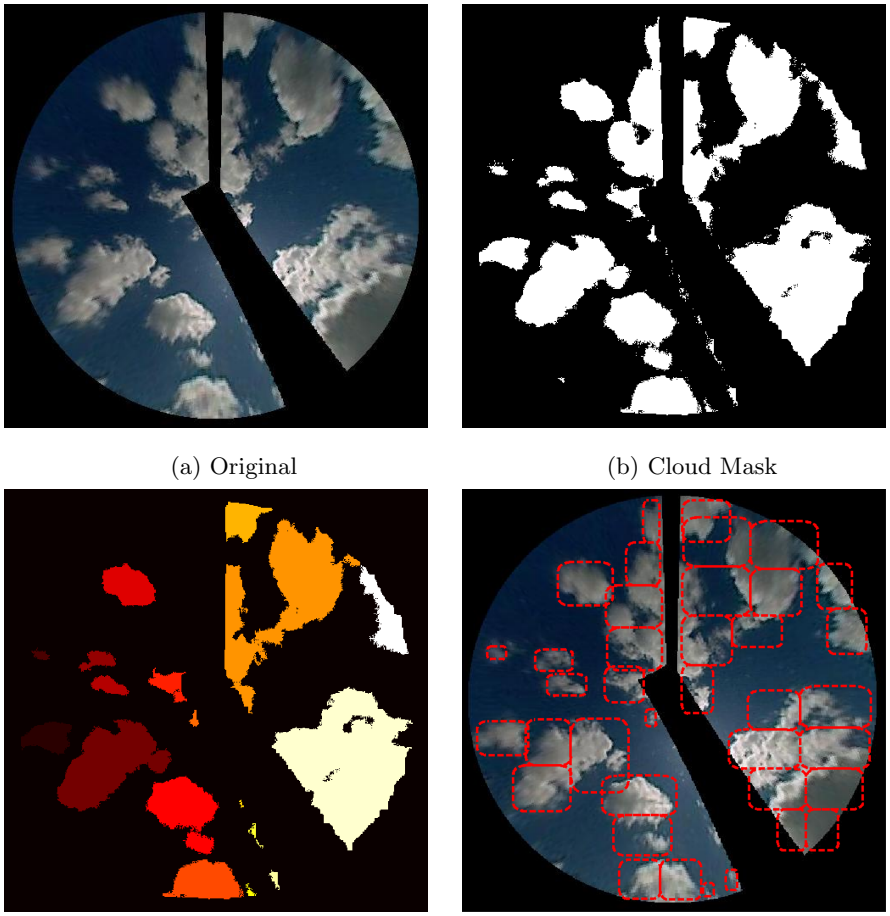
Cloud tracking at a very small scale (e.g., the pixel level) is sensitive to noise and changes in illuminance. Hence, we aggregate the cloud pixels classified by the cloud detection pipeline into cloud blocks, each of which is a unit for cloud tracking. The block-matching algorithm presented in Huang's work [20] is applied to find the best matches for image blocks between two consecutive frames.

Table 1: Overall confusion matrix for the cloud detection pipeline applied to the images shown in Figures 4 and 5. *Left to Right in the Table*: AC_{cld} followed $1 - AC_{cld}$ in the first row and $1 - AC_{sky}$, followed by AC_{sky} in the second row. All values are in [%].

	Cloud Detection Pipeline	
Manual	Cloud	Sky
Cloud	96.6%	3.4%
Sky	10.3%	89.7%

When a suitable block size is chosen, state-of-the-art block-matching methods demonstrate stable performance in estimating cloud motions. Similarly, in our approach, we restrict the range of block size between 10x10 and 80x80 to ensure that each cloud block exhibits adequate textural variability while simultaneously includes a few unrelated pixels.

To aggregate cloud pixels into blocks of an appropriate size, we first apply the connected-component detection algorithm [39] to determine the regions that are distinct from clear sky. As shown in Figure 6c), each component marked with a given color has a high likelihood of belonging to the same piece of cloud. For each component, we draw a minimum bounding rectangle box/block to enclose its cloud pixels. However, these bounding blocks may not be of a suitable size for cloud tracking because of the arbitrary size and shape of the connected components. Therefore, we apply the iterative rectangle split-and-merge algorithm to all components to ensure that the final cloud blocks have acceptable dimensions in accordance with the pre-defined range (i.e., 10x10 to 80x80). As shown in Figure 6d, the final output is a sequence of cloud blocks derived from connected components, each of which is treated as a unit region for calculating the temporal and spatial correlations of its internal cloud pixels.



(a) Original (b) Cloud Mask

(c) Component Map (d) Cloud Blocks

Figure 6: Generating cloud blocks from a cloud mask. (a) Original image. (b) The cloud mask generated from the original image; (c) The output components detected by applying connected components detection are illustrated in different colors. Using a "split and merge" technique, the final cloud blocks are generated as shown in red in (d).

.

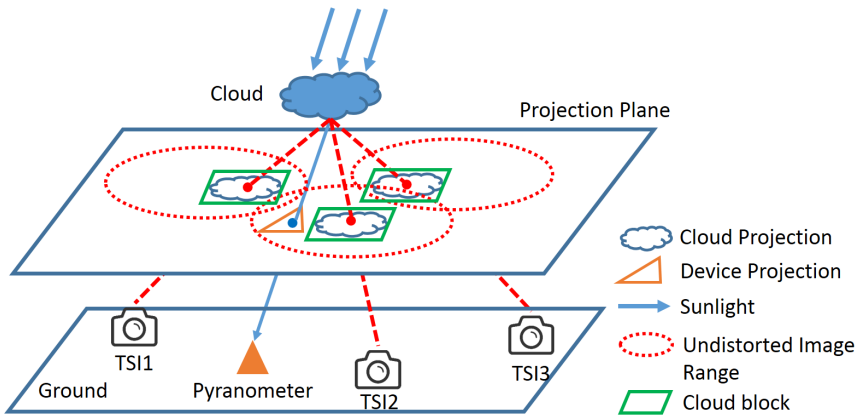


Figure 7: Overview of three-TSI tracking.

4. Cloud Tracking and Layer Determination

4.1. Cloud Base Height and Motion

Using the generated cloud blocks, the basic concept of cloud tracking is to recover the 3D coordinates for each block and estimate their motions. Figure 7 shows an overview of three-TSI tracking. A piece of cloud at a certain level can be projected into/visualized as three different cloud blocks on the projection planes of the three difference TSIs. Because of the geometric difference among the three TSI devices on the ground, the pixel coordinates of each cloud block are distinct. Therefore, these cloud blocks can be combined to recover their 3D coordinates. Theoretically, a pair of TSIs should suffice to recover the cloud base height (CBH) using a stereography approach [40] [41] [42]. In our system, we utilize the redundant information from the three-TSI network to increase the robustness of cloud tracking.

To retrieve the CBH, we define the mathematical term $\mathbf{d}^{i,j,t}$ as the pixel-wise displacement vector between two blocks in different TSI images (TSI_i and TSI_j). If I represents the image matrix of an undistorted image and $\mathbf{c} = (x, y)$ is the center of a cloud block, then the displacement of $TSI_i \rightarrow TSI_j$ at time t

is expressed as $\mathbf{d}^{i,j,t} = (d_x, d_y)$ and satisfies the following:

$$I_i^t(\mathbf{c}) = I_j^t(\mathbf{c} + \mathbf{d}^{i,j,t}) \tag{3}$$

If h^t denotes the CBH detected at time t, based on computational geometry, then there exists a unique mapping f that can be used to calculate h^t from the displacement vector (see Appendix 9.2):

$$h^t = f_{i,j}(\mathbf{d}^{i,j,t}) \tag{4}$$

Conversely, if h^t is calculated from one displacement vector, then all other displacement vectors can be calculated by mapping the derived 3D coordinates back to the projection plane. We define this procedure as a fixed series of transformations g, which are determined only by the locations of the TSI (see Appendix 9.2).

$$\mathbf{d}^{i,j,t} = g_{i,j}(h^t), i, j \in S = \{1, 2, 3\} \tag{5}$$

To track the cloud movement over time, we define another similar term, the motion vector, to describe the pixel-wise shift of a cloud block between consecutive frames. We adopted the traditional Optical Flow (OF) estimation [43] to define the motion vector from the time frame t to the next frame $t+1$ for TSI_i as $\mathbf{v}^t = (v_x^t, v_y^t)$. Assuming consistent brightness, the motion vector satisfies the following equation:

$$I_i^t(\mathbf{c}) = I_i^{t+1}(\mathbf{c} + \mathbf{v}^t) = I_i^{t+1}(x + v_x^t, y + v_y^t) \tag{6}$$

Given the definition of the motion and displacement vectors, we simplify the cloud tracking task by adopting two assumptions: a). clouds exhibit only planar movement, without any vertical motion, and b). the velocity and direction remain constant within our forecasting time window. Moreover, as mentioned in Section 2, all undistorted images reside on the same projection plane, with the identical fields of view and spatial resolutions. Therefore, to ensure the

uniformity of the TSI images and the consistency of the cloud movements without any loss of generality, we make several additional assumptions about our three-TSI system. First, a cloud block and its counterparts in other TSI images have the same dimensions, without scaling or shearing. Figure 7 shows that the cloud blocks obtained from different TSIs to represent the same piece of cloud must be identical in size on the same plane. Second, the mapping from the displacement vector to the CBH is independent of time because the mapping functions f and g are fixed and determined only by the locations of the TSIs (see Appendix 9.2). Third, a cloud block maintains a consistent size over a short time. Because a piece of cloud is assumed to exhibit only planar movement, all its cloud blocks residing on the projection plane must be of the same size at different timestamps. Finally, the fourth assumption is that a cloud block registered by one TSI and its counterparts at the other TSIs have the same motion vector. More importantly, the scale and direction of the motion are consistent, as based on the previous assumption (see Appendix 9.2). We note that our cloud tracking system does not enforce these strong assumptions in practice; however, it remains sufficiently robust to tolerate the differences between the projected images and still identify the same object from different images based on their temporal and spatial correlations.

Based on these assumptions, we designed a novel tracking system utilizing all three TSI devices ($S = \{1, 2, 3\}$) for a series of consecutive timestamps ($T = \{t, t+1, t+2\}$). In total, nine images are used to extract two types of shift vectors at the pixel level: 1) the displacement vectors between TSI views to calculate CBH, and, 2) the motion vectors between consecutive frames from one TSI (planar motion). An intuitive example is illustrated in Figure 8. After identifying a cloud block segment in an image from TSI_i at time t, our goal is to find its "best" matches in the other eight images, and use these matches to explicitly calculate its motion and displacement vectors. On one hand, the motion vectors of the nine cloud blocks should be identical to $\mathbf{v}^t$ because they represent the same cloud with steady movements. Hence, we simplify the tracking problem by setting $\mathbf{v}^t = \mathbf{v}^{t+1}$ during block matching.

Moreover, we assume that the cloud base height remains consistent within a short tracking window. As a result, the estimated CBH derived from the "best" matches must satisfy $h^{t+1} = h^t$, which is also essentially equivalent to $\mathbf{d}^{i,j,t+1} = \mathbf{d}^{i,j,t}$. To identify the best estimates of $\mathbf{v}^t$ and $\mathbf{d}^{i,j,t}$, the next step is to define a criterion to match multiple cloud blocks in different images.

4.2. Similarity Function

For this system, we have designed an algorithm to simultaneously incorporate both spatial and temporal correlations to improve the accuracy of the block-matching methodology. We proposed formulating the tracking problem as the maximization of a similarity function in which these two types of correlation are summed. In detail, this function consists of two components: 1) the sum of the similarity among different time frames for each TSI, and 2). the sum of similarity between any pair of TSIs at a given timestamp. Given a cloud block centered at $\mathbf{c}$ on TSI_1 at time t, the similarity function, φ can be calculated by summing all similarity values, and the motion vector and one displacement vector are denoted by $\mathbf{v}^t$ and $\mathbf{d}^{1,2,t}$ respectively. The formula are as follows:

$$\begin{aligned}\varphi(\mathbf{v}^t, \mathbf{d}^{1,2,t}, \mathbf{c}, t, S) = &\sum_{i,j \in S, i \neq j, t \in T} NCC(I_i^t, I_j^t, \mathbf{c}, \mathbf{d}^{i,j,t}) \\ &+ \sum_{i \in S, t' > t, t' \in T} NCC(I_i^t, I_i^{t'}, \mathbf{c}, (t' - t) * \mathbf{v}^t)\end{aligned} \tag{7}$$

$$\mathbf{d}^{i,j,t} = g_{i,j}(f_{1,2}(\mathbf{d}^{1,2,t})) \tag{8}$$

Note that we include only the displacement vector from TSI_1 to TSI_2 in the similarity calculation. The displacement vectors between any pair of TSIs can be derived similarly using Equation 8. Here, NCC essentially represents the normalized cross correlation between two image blocks of the same size and dimensions, and it is chosen as the criterion for matching image blocks [39]. The

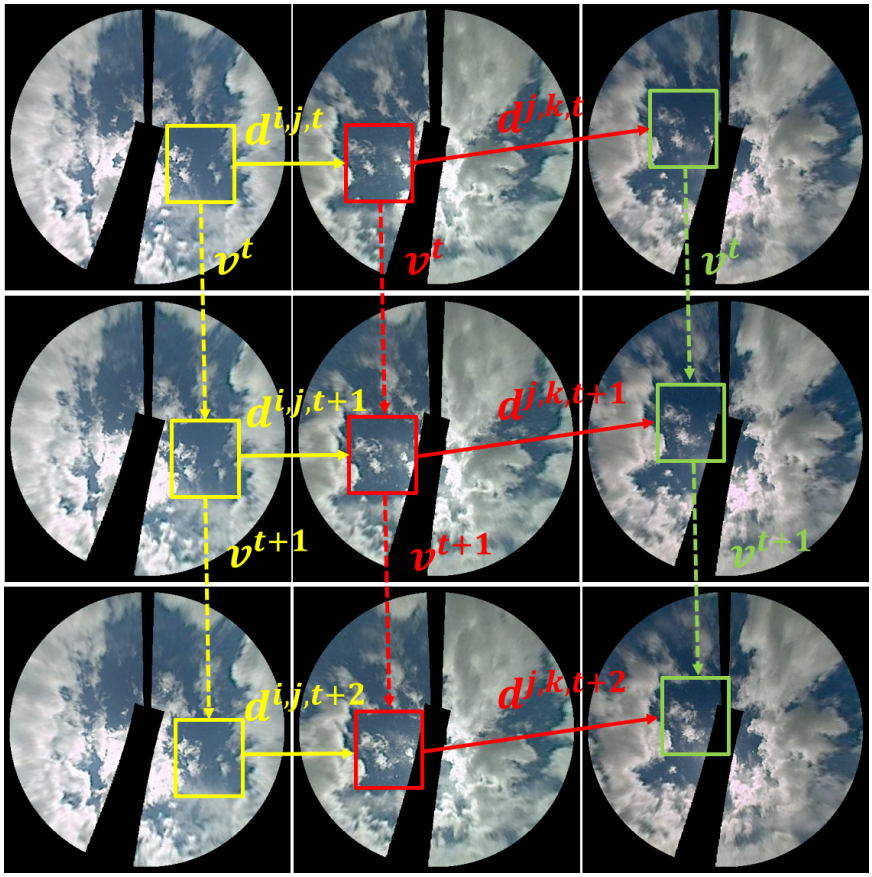


Figure 8: Matching cloud blocks in nine images. The cloud block of interest is indicated in yellow, red, and green boxes on the TSI_1, TSI_2, TSI_3 images, respectively. The movement of the cloud block that is detected between two consecutive frames is indicated by a dotted arrow and labeled as $\mathbf{v}$. The displacement vector between a pair of TSIs at the same timestamp is represented by a solid arrow and labeled as $\mathbf{d}$.

formula for NCC is expressed as follows:

$$NCC(I, J, \mathbf{c}, \mathbf{u}) = \frac{1}{p} \sum_{\mathbf{k}=(i,j)} \frac{\big(I(\mathbf{c}+\mathbf{k}) - \mu(I, \mathbf{c})\big)\big(J(\mathbf{c}+\mathbf{u}+\mathbf{k}) - \mu(J, \mathbf{c}+\mathbf{u})\big)}{\sigma(I, \mathbf{c})\sigma(J, \mathbf{c}+\mathbf{u})} \tag{9}$$

where I and J are the grayscale image matrices converted from the undistorted images. Each pixel value lies in the range $(0, 255)$. Furthermore, $\mathbf{u}$ is a motion or displacement vector. Assuming that the block size is (m,n), then $i \in \{-m/2, -m/2+1, ..., 0, ..., m/2\}$ and $j \in \{-n/2, -n/2+1, ..., 0, ..., n/2\}$ are the pixel indices of the cloud block, $p = m \times n$ is a normalization constant, μ is a mean function, and σ is a standard deviation function of the image block. NCC is widely used in motion estimation because of its simplicity and robustness to noise and changes in intensity.

Given the definition of the similarity function φ, our goal is to search for a combination of ($\mathbf{d}^{1,2,t}$, $\mathbf{v}^t$) that optimizes its value. The most straightforward method is searching all combinations and finding the $(\mathbf{d}_m^{1,2,t}, \mathbf{v}_m^t)$ that maximizes the φ score:

$$(\mathbf{d}_m^{1,2,t}, \mathbf{v}_m^t) = \underset{\mathbf{d}^{1,2,t}, \mathbf{v}^t}{argmax} \ \varphi(\mathbf{v}^t, \mathbf{d}^{1,2,t}, \mathbf{c}, t, S) \tag{10}$$

However, the maximum of φ is only the correct result if a) the cloud block is located in the middle of the FOV and visible in all nine images; b) the image quality is sufficient to accurately display the cloud texture within the block; and, c) the cloud inside the block and its counterparts in the images from the other TSIs have similar shape and size, even from different angles of view. In general, several difficulties prevent us from attaining the optimal solution to maximize φ, as described below. In the previous work of [21], the block tracking near the image boundary was found to be inaccurate because of the loss of information. Only the blocks that are at least a certain distance from the boundary can possibly be matched using NCC value. Moreover, because cloud blocks have various sizes and textures, the tracking performance achieved by applying NCC may vary. In practice, even for cloud blocks that are distributed on the same

layer and exhibit identical movement, their optimal solutions to Equation 10 which should be identical may be similar, but no identical. Another difficulty encountered in cloud-block tracking using TSI images is that some information is missing due to artifacts from the TSI. Blank pixels (marked as black in the TSI images) significantly impact block-matching since the maximization of NCC always favors blocks with lower loss of information. Furthermore, noisy pixels or variations in image brightness may cause instabilities in calculating the NCC score and thereby influence the cloud-tracking performance based on Equation 10. Consequently, the maximum value of φ does not always guarantee the best matches among nine images. In practice, we use $(\mathbf{d}_m^{1,2,t}, \mathbf{v}_m^t)$ only as a reference solution for determining the layer information.

As discussed in the previous section, static features are not effective for differentiating cloud pixels near the sunspot. We propose incorporating the dynamic information from cloud tracking to enhance cloud detection in the sunspot region. To mitigate the influence of false-positive pixels around the sunspot, we apply a pre-defined sunspot mask (window) and acquire the motion vector $\mathbf{v}_m^t$ by solving Equation 10 for each cloud block within this range. If the magnitude of $\mathbf{v}_m^t$ for a cloud block is close to 0, then in the subsequent cloud-layer determination step, we convert the relevant cloud pixels within this block into clear-sky pixels.

4.3. Multi-layer Detection

To locate the best match and find the optimal solution of the similarity function φ, we introduce clustering and multi-layer aggregation during cloud tracking to utilize all cloud blocks in the visible range. Instead of considering only the maximum of φ, we propose tracking all possible $(\mathbf{d}^{1,2,t}, \mathbf{v}^t)$ combinations as potential solutions. In practice, we store a combination for a cloud block only if each NCC score in the φ calculation is above a certain threshold (here, we use 0.5). Therefore, a single block may have multiple motion and displacement vectors. Our goal is to obtain $\mathbf{L}$, a collection of multiple potential

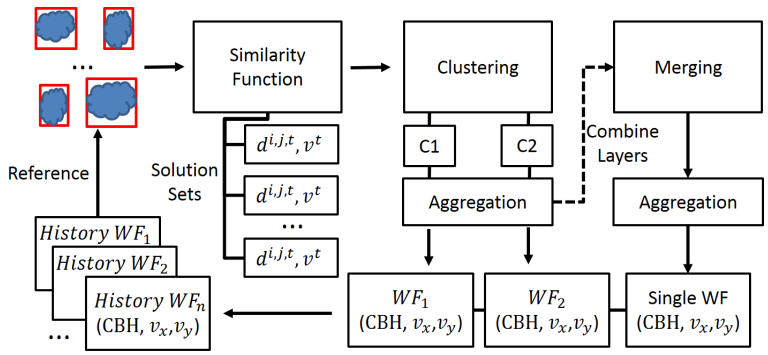


Figure 9: Overview of cloud-block tracking and the determination of multi-layer wind fields.

$(\mathbf{d}^{1,2,t}, \mathbf{v}^t)$ combinations detected for all cloud blocks:

$$\mathbf{L} = \{(\mathbf{d}^{1,2,t}, \mathbf{v}^t) | NCC^* \geq 0.5, \mathbf{c} \in C\} \tag{11}$$

Note that NCC^* refers to any NCC calculation listed in the φ equation. C denotes the set of all cloud blocks detected in the cloud detection pipeline.

Based on $\mathbf{L}$, we apply the k-means clustering [44] technique to generate layers of clouds. In our system, two layers at most are considered and aggregated. Hence, binary clustering is used to split $\mathbf{L}$ into two categories. If we use the clustering result to represent cloud layers, then the centroid of a cluster, $(\mathbf{d}^{1,2,t}, \mathbf{v}^t)$, represents the primary height and motion of this layer. Thereafter, we can group all cloud blocks into two layers/clusters based on the Euclidean distance between the reference solution $(\mathbf{d}_m^{1,2,t}, \mathbf{v}_m^t)$ and the centroids. The entire cloud block set C is then divided into two layers, C_1 and C_2, each of which contains multiple cloud blocks that should have similar motions and heights. To generate more accurate and robust information concerning cloud layers, we assume that cloud blocks on the same layer possess only one major planar motion vector and one CBH. Similar to the form of the centroid in clustering, we define the wind field (WF), $(\mathbf{d}^{1,2,t}, \mathbf{v}^t)$, as the unique combination of the displacement and motion vector corresponding to a given cloud layer. We calculate this combination of vectors by maximizing the summation of φ over all cloud blocks on the same

layer:

$$WF_i = \underset{\mathbf{d}^{1,2,t},\mathbf{v}^t}{argmax} \sum_{\mathbf{c}\in C_i} \varphi(\mathbf{v}^t, \mathbf{d}^{1,2,t}, \mathbf{c}, t, S) \tag{12}$$

If two wind fields extracted from two clusters are sufficiently similar to each other, then we treat them as a single cloud layer in which $\mathbf{c}$ can be drawn from the entire cloud block set C. Similarly, the single wind field can be expressed as follows:

$$WF_{single} = \underset{\mathbf{d}^{1,2,t},\mathbf{v}^t}{argmax} \sum_{\mathbf{c}\in C} \varphi(\mathbf{v}^t, \mathbf{d}^{1,2,t}, \mathbf{c}, t, S) \tag{13}$$

When the Euclidean distance between two motion vectors is less than two pixels, or their height difference is less than 500 meters, we consider these two wind fields to belong to a single layer. To further improve tracking performance and reduce computational complexity, all generated wind fields are stored as historical layers. When searching for a reasonable solution set, this historical reference is used to refine the possible range of motion vectors and height levels. This approach accelerates the search procedure and increases the system's robustness in coping with noise.

An example with two (layers of) wind fields detected is shown in Figures 10 and 11. Nine images in three consecutive frames from timestamp t to $t+2$ are used to extract potential solutions. In Figure 11, the displacement vectors of $TSI_2 \rightarrow TSI_3$ and $TSI_2 \rightarrow TSI_1$ are shown in blue and green respectively, whereas the cloud motions are shown in red. Figures 11a-11c confirm that to maximize φ for each individual block does not always guarantee finding the best matches. We observe that cloud blocks near the shadow band and supporting arm (black area) exhibit obvious errors due to the information loss caused by blank pixels. However, after clustering and aggregating, the cloud blocks can be categorized into two layers, $WF_1 = (1201m, 10px, -6px)$ and $WF_2 = (4184m, 7px, -2px)$. As shown in Figure 11d- 11f, these two layers are indicated by red and green boxes, respectively, and offer more stable estimations of displacement and motion than do the individual cloud blocks.

We also present another example of a single WF detected from the nine input images shown in Figure 8. In Figures 12a- 12c, several combinations of height and motion that achieve the maximum value of φ are deemed outliers and marked with green rectangles. Running the clustering algorithm identified two wind fields: $WF_1 = (4357m, 8px, 8px)$, and $WF_2 = (2098m, 8px, 9px)$. However, these two layers are very similar to each other given their similar motion vectors. Thus, we aggregated all blocks into one layer, as indicated by the red boxes in Figure 12d- 12f. The output WF_{single} is then represented as a single tuple $(2130m, 8px, 9px)$ which is less sensitive to the boundary or noisy cases.

4.4. Multi-TSI View

After determining the cloud layers, multiple TSI views can be stitched together block by block. We select TSI_2 as the origin/center of this combined view because it is located in the middle. The pixels of a cloud block from TSI_1 and TSI_3 can then be mapped to TSI_2's coordinate system through pixel-wise shifts of the forms $TSI_1 \rightarrow TSI_2$ and $TSI_3 \rightarrow TSI_2$ which are identical to the extracted displacement vectors $\mathbf{d^{1,2,t}}$ and $\mathbf{d^{1,3,t}}$. Thus, given the estimated heights of the cloud layers, all cloud blocks from TSI_1 and TSI_3 are placed and stitched into the aggregated view from all three TSIs. An example of such a stitched view wherein all blocks belong to a single layer is shown in Figure 12g. For a case with more than one layer, the stitched view is generated by stitching layers one by one, i.e., from a higher altitude layer to a lower one, because low-altitude clouds can overlay those at higher altitudes. Consequently, the output view should be similar to Figure 11g. The black areas where gaps still exist are marked as blank, and we fill them in with the default sky color for visualization. The default value is calculated by averaging all clear-sky pixels across multiple TSI views.

5. Radiation Prediction Models

In this section, we describe the methods for extracting image features and the different irradiance models used to forecast solar irradiance. First, given the locations of the 25 pyranometers, we need to identify the relevant pixels on the TSI images that are correlated with the surface irradiance fluctuations at these locations. Since the clouds located between the sun and the solar panels are the primary cause of ramps in irradiance, the basic concept of irradiance forecasting is to predict whether clouds will block the sunlight at a specified time. As shown in Figure 7, the device's projection on the sky image corresponds to the intersection of the projected plane (i.e., the image plane) with a line drawn from the sun to the pyranometer. If a cloud is located in the path of the sunlight traveling to the pyranometer, then its projected position on the image plane must overlap with the device's projection. Furthermore, we define the corresponding pixel at the projected position in the stitched view of the three TSIs as a "sun-blocking" pixel. Because this pixel is correlated with the amount of direct sunlight cast on a pyranometer, our goal is then to extract the image features of this pixel which will subsequently serve as the input to the irradiance model. As shown in Figure 7, the position of the sun-blocking pixel depends on the angle of the Sun, the geolocation of the pyranometer, and the height of the cloud that is blocking the direct sunlight. Using these inputs, we can easily calculate the sun-blocking pixel for each of the 25 pyranometers, as shown in Figure 13a and 13b.

To predict which pixels will become sun-blocking pixels in the future, we apply a backtracking method to the current stitched view. In this processing step, the backtracking is guided by the motion vectors detected at previous timestamps: given that the cloud motion in a particular layer is $\mathbf{v^t}$ where t is the current timestamp, the pixel $\hat{\mathbf{c}}^t$ that will potentially become a sun-blocking pixel $\mathbf{c}^{t+N}$ in the Nth future time frame (at time $t+N$) is then calculated by moving $\mathbf{c}^{t+N}$ in the direction opposite to the motion vector $\hat{\mathbf{c}}^t \leftarrow \mathbf{c}^{t+N} - \mathbf{v^t} \times N$. Here, we assume that the motion vector remains unchanged between t and $t+N$

and that the pixel at $\hat{\mathbf{c}}^t$ will become $\mathbf{c}^{t+N}$ and block the sunlight after N time frames, provided that it contains a cloud at that time. If multiple layers of clouds move to this pixel location, then the lower-altitude cloud pixel is preferred because it will block the higher-altitude one.

As expected, the effectiveness identification of the sun-blocking pixel requires an accurate estimation of the motion vectors of each layer. Any error in this information will further accumulate in subsequent steps. To mitigate this effect, we have devised two strategies to be implemented in the feature extraction step. First, instead of focusing on a single sun-blocking pixel, we considered the 7x7 sun-blocking window surrounding this pixel. This enables us to reduce the risk of false prediction, and more importantly, to include more features from the neighboring pixels. Second, we use 23 significant features to describe the spectral variation inside the window at the current time, t and the predicted time, $t + N$. Among these features, nine are the average, minimum, and maximum values of the RGB channels of the window surrounding $\mathbf{c}^t$, which describe the spectral properties of the current observation. Similarly, we choose nine additional features from the sub-blocking window centered at $\hat{\mathbf{c}}^t$, which represent the estimated properties. The remaining five features, are the RBR of $\mathbf{c}^t$ (RBR^t) and $\hat{\mathbf{c}}^t$ ($\hat{RBR}^t$), the cloud fraction at time t, and the ground-truth irradiance values at the current time t (k^t) and one-minute earlier, at $t - 6$ (k^{t-6}). The RBR at both timestamps are used because this quantity shows a noticeable spectral difference between cloud and sky [6]. The cloud fraction represents the overall cloud conditions. The two remaining features, the ground-truth irradiance values at t and $t - 6$, enhance the performance for a short forecasting window because they incorporate persistent observations. Here, the forecasting problem is formulated as $\hat{k}^{t+N} = f(\mathbf{x}^t)$, where $\hat{k}^{t+N}$ represents the predicted irradiance at $t + N$ and $\mathbf{x}^t$ is the vector of the 23 extracted features.

To examine the predictive capabilities of the selected features, we explore four different irradiance models that use a subset or all of these features to generate a regular linear regression or more complicated non-linear relationships $f(\mathbf{x}^t)$: 1) linear RBR delta, 2) an ordinary linear regression model, 3) Support

Vector Regression (SVR) based on a linear kernel, and 4) SVR with a non-linear kernel. For comparison, we use the persistent model as the baseline model, i.e., the **radiation shift**, $RShift$, which directly uses the current observation as the forecasting result. In addition, we introduce one regression model that depends only on multiple irradiance values and is used to further validate image features for prediction capability in our comparison studies.

$$\hat{k}^{t+N} = k^t \tag{14}$$

First, we simply extend the persistent model by incorporating a linear regularization term based on the information from the sun-blocking pixel. Our preliminary study of motion vectors [21] indicates that the RBR of a sun-blocking pixel is a useful indicator of the cloud transmittance at this pixel. The **linear RBR delta** model, denoted by $linear_\delta$, is as follows:

$$\hat{k}^{t+N} = k^t + C \cdot (\hat{RBR}^t - RBR^t) \tag{15}$$

where C is a negative coefficient. We then generalize the model $linear_\delta$ to an **ordinary linear regression model** ($linear_{all}$) over all attributes in $\mathbf{x}_t$:

$$\hat{k}^{t+N} = \mathbf{w} \cdot \mathbf{x}^t + b \tag{16}$$

where $\mathbf{w}$ is the weight vector, and b is the intercept. However, the ordinary model is sensitive to noise or outliers and suffers from the overfitting problem. To overcome these shortcomings, we apply the **Support Vector Regression** (SVR) [45], an extension of the SVM approach, for regression. Given a linear kernel, viz, SVR_{linear}, the radiation estimation still follows the ordinary form given in Equation 16, but the $\mathbf{w}, b$ pair is obtained by solving the classic SVR optimization problem:

$$\min_{\mathbf{w},b,\xi,\xi^*} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{17}$$

subject to

$$\begin{aligned}(\mathbf{w}\cdot\mathbf{x}^t+b)-k^{t+N}&\leq\varepsilon+\xi_t,\xi_t\geq 0,\forall t\\(\mathbf{w}\cdot\mathbf{x}^t+b)-k^{t+N}&\geq-\varepsilon-\xi_t^*,\xi_t^*\geq 0,\forall t\end{aligned}\tag{18}$$

where ε is the regression margin, ξ_t and ξ_t^* are the slack variables, and C is a regularization parameter. By incorporating different kernels, we can extend the SVR to model a non-linear relationship by projecting the current attributes into high-dimensional transformed spaces, for example, Fourier space. To this end, we use the Radial Basis Function (RBF) kernel, SVR_{rbf}, with σ as the RBF smoothing parameter:

$$\begin{aligned}\hat{k}^{t+N}&=\kappa(\mathbf{w}',\mathbf{x}^t)+b\\&=e^{-\frac{\|\mathbf{w}'-\mathbf{x}^t\|^2}{2\sigma^2}}+b\end{aligned}\tag{19}$$

For both the linear and non-linear SVR models, certain parameters, such as, the regularization weight C, the error tolerance range ε and the RBF smoothness factor σ, must be tuned to achieve the optimal performance. For our experiment, we selected the optimal parameters from within certain ranges. Here, C is taken from $C=[-1024,1024]$, ε from $\varepsilon=[0.001,0.6]$, and σ from $\sigma=[10^{-4},10^4]$. The average values of the 25 solar sensors for SVR_{linear} are set as follows: $C_{avr}=35,\varepsilon_{avr}=0.01$. In the case of SVR_{rbf}, $C_{avr}=19$, $\varepsilon_{avr}=0.003$, and $\sigma_{avr}=0.2291$. We trained the $linear_{all}$ and $linear_{\delta}$ models by minimizing their least-square errors. The persistent model $RShift$ requires no training.

6. Forecasting Model Results

6.1. Experimental Dataset

We chose the period from May 13, 2013, to June 03, 2013, encompassing various weather conditions, to evaluate the performance of 1-min- to 15-min-ahead 3D cloud tracking and irradiance forecasting. To guarantee consistent cloud visibility and zenith angle, we filtered out records whose timestamps are not between 9:00AM and 16:00PM Eastern Standard Time (EST). Our experimental dataset contains 9963x3 images from three TSIs that collect data simultaneously

and the synchronized 9963x25 GHI records from 25 ground-based pyranometers. Furthermore, we divided the full dataset into four categories based on weather and cloud conditions: single-layer clouds (D_s), multi-layer clouds (D_m), overcast or extremely cloudy (D_o), and a mixture thereof (D_{mix}). D_s corresponds to a typical low-altitude cloud that is commonly observed on the East Coast of the United States. This type of low cloud typically appears in a single layer and remains in the field of view of a TSI for several minutes at most. Given its quick movement in and out of the FOV of a TSI, D_s is an ideal dataset for demonstrating the capability of the new tracking system to capture rapid cloud motion and evolution. D_m is a collection of cases wherein multiple layers of clouds were observed within the FOV of all TSIs. Hence, the tracking and forecasting performance with respect to D_m is expected to reflect the ability of the system to categorize and track multiple wind fields. D_o corresponds to cloudy and overcast cases. Based on our observations, the textural patterns of the images acquired in extremely cloudy conditions are not obvious and are hence difficult to utilize for block-matching and the extraction of image features. Therefore, we designed D_o to demonstrate the sensitivity of this new tracking system when applied to TSI images with less obvious patterns. D_{mix} contains the remaining cases, including sunny, partially cloudy, and thin-layer conditions. The cloud tracking and irradiance forecasting performance with respect to D_{mix} reflects the system's average performance for a mixture of cloud conditions. Table 2 describes the details of all subsets.

6.2. Evaluation Metrics and Validation Method

As discussed in Section 5, the prediction of the sun-blocking pixels for the 25 pyranometers is performed based on cloud movements in a particular cloud layer. When a cloud moves too rapidly, or the forecasting interval is too long, one or more positions may be outside of the FOV of our TSI. In this case, we cannot extract any features for modeling and predicting irradiance. To quantify the tracking capability for sun-blocking pixels within our stitched view, we define a metric named the Successful Tracking Index (STI), i.e., the percentage of data

instances in the designated dataset for which all 25 sun-blocking pixels can be included in the FOV of the stitched image for a particular forecasting horizon. The STI is calculated as follows:

$$STI = \frac{N - N_{oof}}{N} \tag{20}$$

Here, N is the total size of the dataset, while N_{oof} represents the number of records for which one or more sun-blocking pixels lie outside the field of view.

We also use the mean absolute error (MAE) and root-mean-square-error (RMSE) as evaluation metrics. In our experiment, the MAE measures the average accuracy of cloud tracking, whereas the RMSE assigns greater penalties to large errors, such as falsely estimation of a cloud's presence. To avoid bias and control the over-fitting problem, we introduce the cross-validation technique [35] into the modeling and evaluate the forecasting performance across all 25 stations. In the cross-validation, the original dataset is evenly divided into several independent subsets, and the average performance of the predictive model measured across these subsets. In this experiment, five-fold cross-validation ($N_{cv} = 5$) is applied, the five folds (four for training and one for testing) randomly generated. Additionally, since 25 simultaneous, location-dependent ground measurements are available ($s = 1, 2, ...25$), we can verify a forecast by comparing it with the observations from a different measurement station, s, at time t. Hence, the final error metrics can be formulated as the average performance of all five-fold tests across all 25 stations as follows:

$$MAE = \frac{1}{N_{cv}} \sum_{i=1}^{N_{cv}} \frac{1}{N_i} \sum_{s,t} \left| k_s^t - \hat{k_s^t} \right| \tag{21}$$

$$RMSE = \frac{1}{N_{cv}} \sum_{i=1}^{N_{cv}} \sqrt{\frac{1}{N_i} \sum_{s,t} (k_s^t - \hat{k_s^t})^2} \tag{22}$$

Here, N_i is the size of the test fold, i, generated in the cross-validation. The MAE and RMSE score are calculated based on the normalized GHI values.

To evaluate the effectiveness of the image features extracted from our multi-layer cloud detection and tracking system, we created a new reference model,

SVR_k, for comparison. This model also uses SVR with only radiation observations as inputs and excludes all image-based features, in contrast to our proposed forecast model SVR_{rbf}. In greater detail, SVR_{rbf} and SVR_k share the same radial basis function as their SVR kernel and use identical parameter settings for both training and testing. The difference is that SVR_k uses only the l (here, we choose $l = 6$) most recent radiation values, i.e., $k^{t-5}, k^{t-4}...k^t$, as inputs. We apply both SVR_{rbf} and SVR_k to generate real-time irradiance predictions to validate the effectiveness of the image features in producing short-term solar forecasts.

6.3. Model Performance

Figure 14 shows the 1-15 minute STI scores of the entire dataset (bars) and the four subsets (lines). We observe that the STI scores for D_s and D_m decrease dramatically beginning with the nine-minute forecast, whereas the performance for the other two subsets remain relatively stable between one and fifteen minutes. Consequently, the success ratio for the entire dataset, *overall*, also decreases as the time horizon increases. The STI decreases for longer forecast horizons (longer than ten minutes) because of fast-moving clouds. According to our observations from the TSI images, on the East Coast of United States, these clouds are mostly distributed in a single layer with a cloud base height below 3000 meters. They often have a high velocity, as detected at the pixel level, and exhibit rapid formation/dissipation within a 10-minute window. Thus, given the limited visible range and TSI resolution, 3D cloud tracking can capture the majority of low-layer clouds for only up to nine minutes. Beginning at the 10-minute horizon, low clouds are highly likely to move out of the field of our stitched view. This cloud property, combined with the physical limitations of the tracking TSIs is consistent with the significant decrease in the STI of D_s at the nine-minute horizon, reaching almost 0 at the ten-minute horizon. Similarly, the STI of D_m decreases after nine minutes. However, since more than one layer is present in D_m, the tracking results benefit from partial estimations from the higher layers which tend to be more stable and have slower pixel-wise motion

vectors. For D_o and D_{mix}, most cases correspond to high-layer clouds or mixed conditions and are therefore suitable for ten-minute forecasts and beyond.

We evaluated the irradiance forecasting performance over the entire dataset based on the metrics of the MAE and RMSE scores, and the results are presented in Tables 3 and 4. We excluded the out-of-FOV data points and only trained and tested the irradiance models based on the remaining available data subset. During model training, we discovered that when STI is low, the models tend to overfit the data. One reason for this behavior is that an excess of out-of-FOV records leads to a lack of observations in the training folds. For instance, if we train the model using D_s for forecast horizons longer than ten minutes, we barely have enough training records to generate the forecasting model. Another reason is that we may introduce bias into the forecasting models. Since rapid-changing cases, such as those with low-altitude clouds, are excluded for long forecasting horizons, the forecasting models will place more weight on the "easy" cases, such as those corresponding to sunny and overcast conditions. Hence, if too many records in an experimental dataset are out of the FOV, or the STI value is below a certain threshold (here, 60%), we mark the result with an asterisk to indicate a partial forecast and a potential overfitting problem. When all records are out of the FOV ($STI = 0$), we denote this scenario with '-', indicating that no forecasting result is available (see Table 4).

In Figures 15a and 15b, we evaluate the effectiveness of the four forecasting models by comparing them with the persistent model for one- to fifteen-minute forecasts. To ensure that the sun-blocking pixels are contained within the stitched view for the majority of the training and test cases over the full forecast horizons, we use the experimental dataset D_{mix} which has the most stable STI values (Figure 14). Figures 15a and 15b show that $linear_\delta$ consistently acquires less large forecast errors in comparison with the persistent model (as measured by RMSE), but exhibits the worst performance in terms of average accuracy of irradiance forecast (as measured by MAE), confirming that tracking a single sun-blocking pixel leads to a high risk of deviating from its real position and thereby falsely predicting the presence of clouds. By virtue

of our proposed feature extraction in the sun-blocking window, this problem is mitigated in the other three models which consistently provide better results in both metrics. In comparison to $linear_{all}$ which was trained using least-square errors, the linear SVR approach, SVR_{linear}, tends to be more robust in terms of average errors (MAE); however, it is very sensitive to large errors (RMSE). The proposed non-linear model, SVR_{rbf}, effectively addresses this problem by mapping non-linear relations to linear ones in a higher dimensional space. As is evident from these two figures, SVR_{rbf} effectively reduces the occurrence of large errors and outperforms the other four models in terms of the RMSE and MAE metrics.

We also analyzed the performance of SVR_{rbf} on the entire dataset and compared it with $RShift$ across all 25 ground measurements. Figure 16 shows the percentage of reduction in MAE achieved using SVR_{rbf}. The blue shaded regions represent the upper and lower improvement bounds for all 25 pyranometers, whereas the mean improvements are plotted as blue dots with standard deviation bars. We observe that beyond a nine-minute horizon, the uncertainty in performance improvement, as indicated by the upper and lower bounds, increases with the increase in forecasting horizon. This is expected because many data points are out-of-FOV for the D_s and D_m subsets, which affects the reliability beyond the nine-minute horizon. Moreover, several successful cases that remain stable even beyond the nine-minute forecast are "easy" ones, such as sunny and overcast conditions. Therefore, the persistent model $RShift$ can take advantage of these cases to minimize error and is thus difficult to outperform. In the same plot, we observe that both the difference between the upper and lower bounds and the standard deviation increase as the time horizon increases. We observe that despite the expected uncertainties arising for a long-term forecast, the SVR_{rbf} model is nevertheless significantly superior to the persistent model, achieving at least a 26% improvement.

To apply this study to real-time forecasting, we averaged the SVR_{rbf} parameters over the entire dataset and constructed the radiation-only model SVR_k to generate 5-, 10-, and 15-minute forecasts for May 14, 2013. Figure 17 compares

the forecasting results of SVR_{rbf} and SVR_k with the real measurements from the deployed pyranometers. In the individual figures, the normalized GHI predictions are converted back to real values using Equation 1. To be consistent with the irradiance plots in Figure 17, the root-mean-square errors (RMSE) in the caption are calculated directly from ground-truth measurements (real GHI values) without normalization. The gray/dark areas in these figures represent gaps with no prediction which include the cases corresponding to the low zenith angles during the early morning and late afternoon as well as the periods in which sun-blocking pixels were out of the FOV. The results show that the five-minute forecasts generated using SVR_{rbf} achieve good accuracy and capture most radiation ramps. Meanwhile, the number of detected ramps decreases for ten-minute and fifteen-minute forecasts because of the instability of the motion vectors and the occurrence of clouds that may reside outside the field of the stitched view. Moreover, because SVR_k only relies on radiation features, we observe that it exhibits a behavior similar to that of the persistent model, often failing to detect radiation fluctuations and generating false alarms based on previous irradiance trends. For a longer forecasting horizon, such as 10 or 15 minutes, the forecasting accuracy of SVR_k decreases rapidly, and the model cannot faithfully predict irradiance ramp events. Compared with SVR_k, SVR_{rbf} incorporates multiple features derived from predicted cloud movements and sky images and consequently, introduces fewer forecasting errors and captures more ramp events.

7. Conclusions

In this paper, we proposed a novel 3D cloud detection and tracking system using computational geometry and machine learning techniques. By taking advantage of redundant information from the overlapping views of multiple cameras, this system can not only identify clouds at the pixel level, but also determine their base heights and the wind fields of multiple cloud layers. This information can be used to stitch multiple TSI views together to generate an

expanded cloud view that is larger than that obtained from a single camera, thereby increasing the forecast horizon. More importantly, based on the predicted layers and the stitched view, we investigated the use of image-based features and irradiance models to effectively capture short-term fluctuations in solar irradiance.

To verify the effectiveness of our tracking system, we organized our experimental data into four categories based on cloud conditions and tested the system over 25 ground sites equipped with pyranometers. The results showed that our new cloud tracking system provided reliable fifteen-minute forecasts for all 25 ground stations under most cloud conditions. Even for low-altitude and fast-moving clouds, this system can obtain reasonable estimates for up to nine minutes ahead. Furthermore, in our cloud tracking system, all four proposed irradiance models considerably reduced the prediction errors for forecasts of up to fifteen minutes ahead. Compared with the persistent model, both linear and non-linear models based on the extracted sky-image features significantly offered significantly improved accuracy with respect to the MAE and RMSE metrics.

In the future, we will explore new techniques for overcoming limitation in the FOVs of observations and introduce more useful features into the forecasting models. Although the stereo stitching of sky images from multiple TSI sources enlarges the visible range, it can only expand the forecasting range to a certain degree. A possible solution to this problem is to explore the statistical properties that can be derived from feature information in the cloud detection and tracking steps and to perform statistical estimations of the out-of-FOV cases. Moreover, our experiment demonstrated that individual image-scale features are prone to error accumulation over time. We will explore additional features from sky images and other sources, such as, LIDAR, temperature, and past predictions from the model itself, to improve its robustness and reduce forecasting errors.

8. Acknowledgment

This research receives the generous support from the DOE project "A Public-Private-Academic Partnership to Advance Solar Power Forecasting". It is partially supported by DOE grant DE-AC02-98CH10886.

9. Appendix

9.1. Assumptions of Consistency of Projected Images Across TSIs

The vertical height and horizontal expansion of clouds vary considerably among different types of clouds. Therefore, the projected size of a cloud observed by ground-based cameras depends strongly on the cloud type, the location of each camera, the field of view, and the solar zenith angle. With no loss of generality, we assume that the dimensions of the projected views of the cloud base in images from multiple TSIs are identical to ensure the suitability and mathematical correctness of the following calculations. In practice, our cloud tracking system does not enforce these strong assumptions; however, it is sufficiently robust to tolerate the differences between projected images while still being able to identify the same object in different images based on the values of the temporal and spatial correlations.

9.2. Consistency of Motion and Displacement Vectors

In this section, we assume that the motion vectors are consistent in spatial representation across all TSIs and that the displacement vector $\mathbf{d}^{i,j,t}$ is uniquely mapped to the cloud base height h^t. The pixel coordinates, (x_i, y_i), for TSI_i can be transformed into a coordinate system relative to the center of the camera (x_{ci}, y_{ci}). The azimuth and zenith angles (A_i and Z_i) at this point are represented using the image coordinates as follows:

$$rx_i = x_i - x_{ci} \ , \ ry_i = y_{ci} - y_i \tag{23}$$

$$Z_i = \arctan \frac{\sqrt{(rx_i^2 + ry_i^2)} \times \tan\theta_i}{R_i} \tag{24}$$

$$A_i = \arctan \frac{rx_i}{ry_i} \tag{25}$$

θ_i and R_i are pre-defined parameters in preprocessing to control FOV of the undistorted image for TSI_i. θ_i stands for the max zenith angle of FOV while R_i is the radius of visible scope at pixel level. Given the base height h, we can recover the 3-D spatial coordinates (sx_i, sy_i, sz_i) relative to TSI_i by

$$sx_i = h \times \tan Z_i \times \sin A_i \tag{26}$$

$$sy_i = h \times \tan Z_i \times \cos A_i \tag{27}$$

$$sz_i = h \tag{28}$$

Then, mapping from the image coordinates to the spatial coordinates can be expressed as follows:

$$sx_i = \frac{h \times tan\theta_i}{R_i} \times (x_i - x_{ci}) \tag{29}$$

$$sy_i = \frac{h \times tan\theta_i}{R_i} \times (y_{ci} - y_i) \tag{30}$$

If the geo-difference between TSI_j to TSI_i is defined as $(D_{i,j,x}, D_{i,j,y})$, then the same object in spatial coordinates system relative to TSI_j can be acquired through the following equation:

$$sx_j = sx_i + D_{i,j,x}\ ,\ sy_j = sy_i + D_{i,j,y}\ ,\ sz_j = sz_i \tag{31}$$

Combine Equation 29, 30 and 31, we can derive image coordinates $(x_j,\ y_j)$ to be:

$$x_j = \frac{R_j \times \tan\theta_i}{R_i \times \tan\theta_j} \times (x_i - x_{ci}) + \frac{R_j \times D_{i,j,x}}{h \times \tan\theta_j} + x_{cj} \tag{32}$$

$$y_j = y_{cj} - \frac{R_j \times \tan\theta_i}{R_i \times \tan\theta_j} \times (y_{ci} - y_i) - \frac{R_j \times D_{i,j,y}}{h \times \tan\theta_j} \tag{33}$$

Since the preprocessing of undistorting images uses the same scale and view range for all TSIs, from Equation 32 and 33, we derive the following:

$$\Delta x_i = \Delta x_j \ , \ \Delta y_i = \Delta y_j \ , \ when \ \theta_i = \theta_j \ , \ R_i = R_j \tag{34}$$

where Δx_i and Δy_i stand for the offset (distance) between pixel $\mathbf{c} = (x_i, y_i)$ and pixel $\mathbf{c}' = (x_i{}', y_i{}')$ along the x and y direction respectively in an image of TSI_i. Based on our assumptions and Equation 34, its projections on multiple TSIs have an identical size. Moreover, its pixel-wise motion is also the same since it stands for the same scale of movements on the image plane.

Following the definition in Section 4.1, we define the $\mathbf{d}^{i,j,t} = (d_x^{i,j,t}, d_y^{i,j,t})$ as the displacement vector from TSI_i to TSI_j. Based on the result in Equation 34, we acquire the height-to-displacement mapping as follows:

$$d_x^{i,j,t} = x_j - x_i = x_{cj} - x_{ci} + \frac{R_j D_{i,j,x}}{h^t \times \tan\theta_j} \tag{35}$$

$$d_y^{i,j,t} = y_j - y_i = y_{cj} - y_{ci} - \frac{R_j D_{i,j,y}}{h^t \times \tan\theta_j} \tag{36}$$

Therefore, the cloud base height, h^t, can be represented by the displacements in both the x and y directions respectively:

$$h^t = \frac{D_{i,j,x} R_j}{(d_{i,j,x} + x_{ci} - x_{cj})\tan\theta_j} \tag{37}$$

$$h^t = \frac{D_{i,j,y} R_j}{(-d_{i,j,x} - y_{ci} + y_{cj})\tan\theta_j} \tag{38}$$

Equation 37 and 38 should have the same output. Otherwise the displacement vector is invalid since it does not satisfy the spatial representation. The two functions $f_{i,j}$ and $g_{i,j}$ used in Sections 4.1 and 4.2 can be obtained: $f_{i,j}$ is one of Equations 37 and 38 while $g_{i,j}$ is chosen from Equations 35 and 36. Therefore we conclude that given a pair of displacement vector $\mathbf{d}^{i,j,t}$, h^t is uniquely determined by $f_{i,j}$. More importantly, the mapping function series, f and g, are time-independent.

References

[1] H. Escrig, F. Batlles, J. Alonso, F. Baena, J. Bosch, I. Salbidegoitia, J. Burgaleta, Cloud detection, classification and motion estimation using geostationary satellite imagery for cloud cover forecast, Energy 55 (2013) 853–859.

[2] J. Schmetz, K. Holmlund, J. Hoffman, B. Strauss, B. Mason, V. Gaertner, A. Koch, L. Van De Berg, Operational cloud-motion winds from meteosat infrared images, Journal of Applied Meteorology 32 (7) (1993) 1206–1225.

[3] P. Héas, E. Mémin, Three-dimensional motion estimation of atmospheric layers from image sequences, Geoscience and Remote Sensing, IEEE Transactions on 46 (8) (2008) 2385–2396.

[4] J. Bosch, J. Kleissl, Cloud motion vectors from a network of ground sensors in a solar power plant, Solar Energy 95 (2013) 13–20.

[5] C. Long, D. Slater, T. P. Tooman, Total sky imager model 880 status and testing results, Pacific Northwest National Laboratory, 2001.

[6] V. Morris, Total sky imager (tsi) handbook, Tech. rep., ARM TR-017 (2005).

[7] G. Pfister, R. McKenzie, J. Liley, A. Thomas, B. Forgan, C. N. Long, Cloud coverage based on all-sky imaging and its impact on surface solar irradiance, Journal of Applied Meteorology 42 (10) (2003) 1421–1434.

[8] J. Shields, R. Johnson, T. Koehler, Automated whole sky imaging systems for cloud field assessment, in: Fourth Symposium on Global Change Studies, American Meteorological Society, 1993, pp. 228–231.

[9] M. Ghonima, B. Urquhart, C. Chow, J. Shields, A. Cazorla, J. Kleissl, A method for cloud detection and opacity classification based on ground based sky imagery, Atmospheric Measurement Techniques 5 (11) (2012) 2881–2892.

[10] C. N. Long, J. M. Sabburg, J. Calbó, D. Pagès, Retrieving cloud characteristics from ground-based daytime color all-sky images, Journal of Atmospheric and Oceanic Technology 23 (5) (2006) 633–652.

[11] M. Souza-Echer, E. Pereira, L. Bins, M. Andrade, A simple method for the assessment of the cloud cover state in high-latitude regions by a ground-based digital camera, Journal of Atmospheric and Oceanic Technology 23 (3) (2006) 437–447.

[12] Z. Peng, S. Yoo, D. Yu, D. Huang, P. Kalb, J. Heiser, 3d cloud detection and tracking for solar forecast using multiple sky imagers, in: Proceedings of the 29th Annual ACM Symposium on Applied Computing, ACM, 2014, pp. 512–517.

[13] K. Buch, C.-H. Sun, L. Thorne, Cloud classification using whole-sky imager data, in: of the 5th Atmospheric Radiation Measurement Science Team Meeting, San Diego, CA, USA, 1995, pp. 19–23.

[14] M. Singh, M. Glennen, Automated ground-based cloud recognition, Pattern analysis and applications 8 (3) (2005) 258–271.

[15] J. Calbo, J. Sabburg, Feature extraction from whole-sky ground-based images for cloud-type recognition, Journal of Atmospheric and Oceanic Technology 25 (1) (2008) 3–14.

[16] A. Heinle, A. Macke, A. Srivastav, Automatic cloud classification of whole sky images, Atmospheric Measurement Techniques Discussions 3 (1) (2010) 269–299.

[17] A. Kazantzidis, P. Tzoumanikas, A. Bais, S. Fotopoulos, G. Economou, Cloud detection and classification with the use of whole-sky ground-based images, Atmospheric Research 113 (2012) 80–88.

[18] C. W. Chow, B. Urquhart, M. Lave, A. Dominguez, J. Kleissl, J. Shields, B. Washom, Intra-hour forecasting with a total sky imager at the uc san diego solar energy testbed, Solar Energy 85 (11) (2011) 2881–2893.

[19] H. Huang, S. Yoo, D. Yu, D. Huang, H. Qin, Cloud motion detection for short term solar power prediction, in: Proceedings of ICML 2011 Workshop on Machine Learning for Global Challenges., 2011.

[20] H. Huang, S. Yoo, D. Yu, D. Huang, H. Qin, Correlation and local feature based cloud motion estimation, in: Proceedings of the Twelfth International Workshop on Multimedia Data Mining, ACM, 2012, pp. 1–9.

[21] H. Huang, J. Xu, Z. Peng, S. Yoo, D. Yu, D. Huang, H. Qin, Cloud motion estimation for short term solar irradiation prediction, in: Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on, IEEE, 2013, pp. 696–701.

[22] P. Wood-Bradley, J. Zapata, J. Pye, Cloud tracking with optical flow for short-term solar forecasting, in: 50th conference of the australian solar energy society, Melbourne, 2012.

[23] D. Bernecker, C. Riess, E. Angelopoulou, J. Hornegger, Towards improving solar irradiance forecasts with methods from computer vision, in: Computer Vision in Applications Workshop, 2012.

[24] Z. Peng, S. Yoo, D. Yu, D. Huang, Solar irradiance forecast system based on geostationary satellite, in: Smart Grid Communications (SmartGridComm), 2013 IEEE International Conference on, IEEE, 2013, pp. 708–713.

[25] S. Golemati, J. S. Stoitsis, A. Gastounioti, A. C. Dimopoulos, V. Koropouli, K. S. Nikita, Comparison of block matching and differential methods for motion analysis of the carotid artery wall from ultrasound images, Information Technology in Biomedicine, IEEE Transactions on 16 (5) (2012) 852–858.

[26] S. Quesada-Ruiz, Y. Chu, J. Tovar-Pescador, H. Pedro, C. Coimbra, Cloud-tracking methodology for intra-hour dni forecasting, Solar Energy 102 (2014) 267–275.

[27] Y. Chu, H. T. Pedro, C. F. Coimbra, Hybrid intra-hour dni forecasts with sky image processing enhanced by stochastic learning, Solar Energy 98 (2013) 592–603.

[28] R. Marquez, V. G. Gueorguiev, C. F. Coimbra, Forecasting of global horizontal irradiance using sky cover indices, Journal of Solar Energy Engineering 135 (1) (2013) 011017.

[29] J. Shields, M. Karr, T. Tooman, D. Sowle, S. Moore, The whole sky imager–a year of progress, in: Eighth Atmospheric Radiation Measurement (ARM) Science Team Meeting, Tucson, Arizona, 1998, pp. 23–27.

[30] H. Yang, B. Kurtz, D. Nguyen, B. Urquhart, C. W. Chow, M. Ghonima, J. Kleissl, Solar irradiance forecasting using a ground-based sky imager developed at uc san diego, Solar Energy 103 (2014) 502–524.

[31] B. Urquhart, B. Kurtz, E. Dahlin, M. Ghonima, J. Shields, J. Kleissl, Development of a sky imaging system for short-term solar power forecasting, Atmospheric Measurement Techniques Discussions 7 (5) (2014) 4859–4907.

[32] Q. Li, W. Lu, J. Yang, A hybrid thresholding algorithm for cloud detection on ground-based color images, Journal of atmospheric and oceanic technology 28 (10) (2011) 1286–1296.

[33] A. Cazorla, F. Olmo, L. Alados-Arboledas, Using a sky imager for aerosol characterization, Atmospheric Environment 42 (11) (2008) 2739–2745.

[34] T. Lindeberg, Detecting salient blob-like image structures and their scales with a scale-space primal sketch: a method for focus-of-attention, International Journal of Computer Vision 11 (3) (1993) 283–318.

[35] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Ijcai, Vol. 14, 1995, pp. 1137–1145.

[36] M. Stokes, M. Anderson, S. Chandrasekar, R. Motta, A standard default color space for the internetsrgb, 1996, URL http://www. w3. org/Graphics/Color/sRGB.

[37] B. E. Boser, I. M. Guyon, V. N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, ACM, 1992, pp. 144–152.

[38] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[39] R. M. Haralock, L. G. Shapiro, Computer and robot vision, Addison-Wesley Longman Publishing Co., Inc., 1991.

[40] E. Kassianov, C. N. Long, J. Christy, Cloud-base-height estimation from paired ground-based hemispherical observations, Journal of Applied Meteorology 44 (8) (2005) 1221–1233.

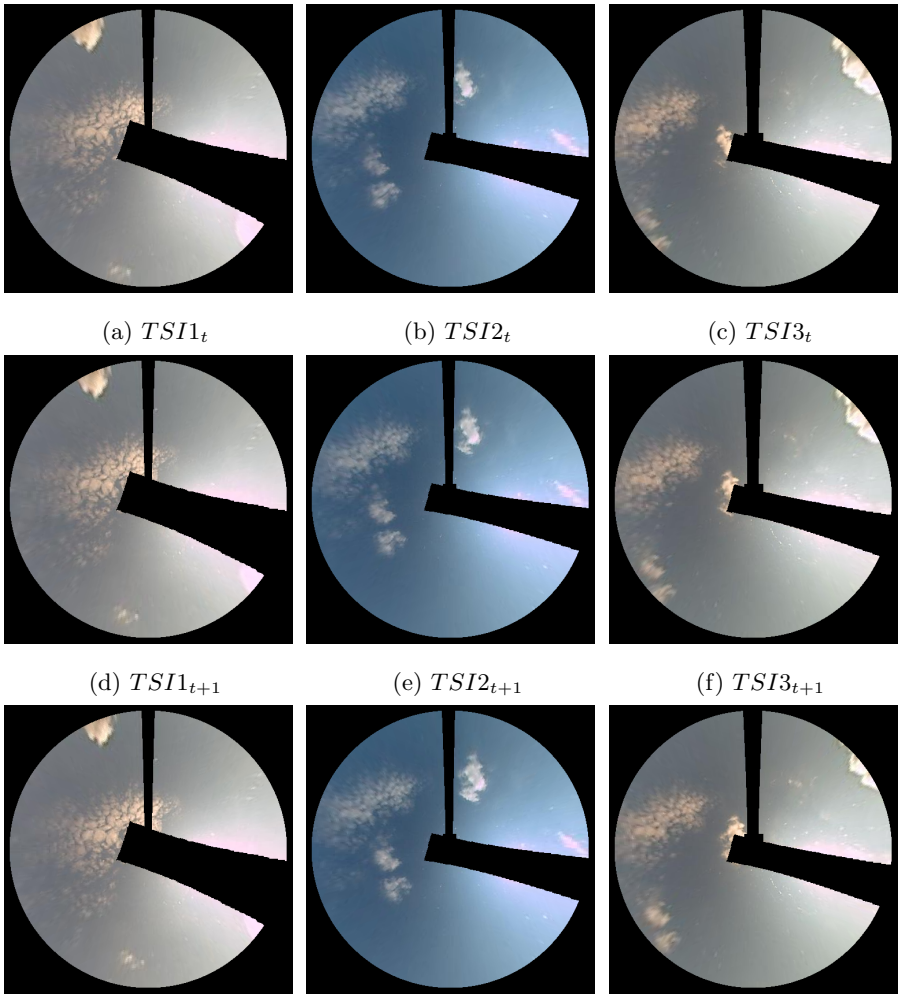
[41] M. C. Allmen, W. P. Kegelmeyer Jr, The computation of cloud-base height from paired whole-sky imaging cameras, Journal of Atmospheric and Oceanic Technology 13 (1) (1996) 97–113.

[42] D. A. Nguyen, J. Kleissl, Stereographic methods for cloud base height determination using two sky imagers, Solar Energy 107 (2014) 495–509.

[43] S. S. Beauchemin, J. L. Barron, The computation of optical flow, ACM Computing Surveys (CSUR) 27 (3) (1995) 433–466.

[44] S. Lloyd, Least squares quantization in pcm, Information Theory, IEEE Transactions on 28 (2) (1982) 129–137.

[45] A. J. Smola, B. Schölkopf, A tutorial on support vector regression, Statistics and computing 14 (3) (2004) 199–222.

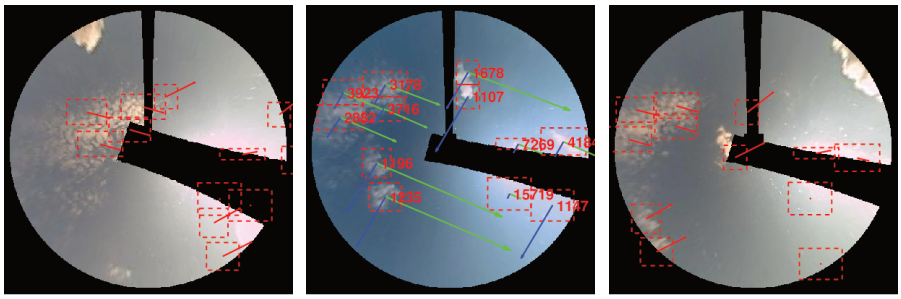


(a) $TSI1_t$ (b) $TSI2_t$ (c) $TSI3_t$

(d) $TSI1_{t+1}$ (e) $TSI2_{t+1}$ (f) $TSI3_{t+1}$

(g) $TSI1_{t+2}$ (h) $TSI2_{t+2}$ (i) $TSI3_{t+2}$

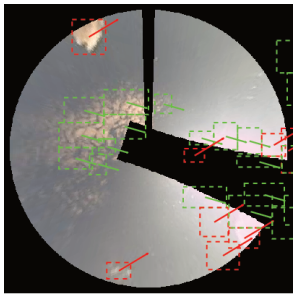
Figure 10: Example of cloud tracking on nine images.



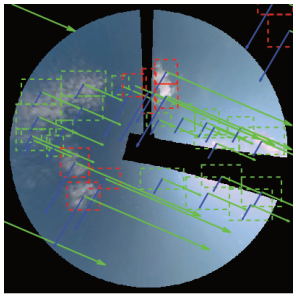
(a) Max. φ at $TSI1_t$

(b) Max. φ at $TSI2_t$

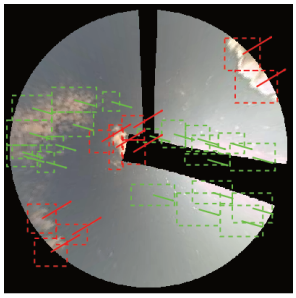
(c) Max. φ at $TSI3_t$



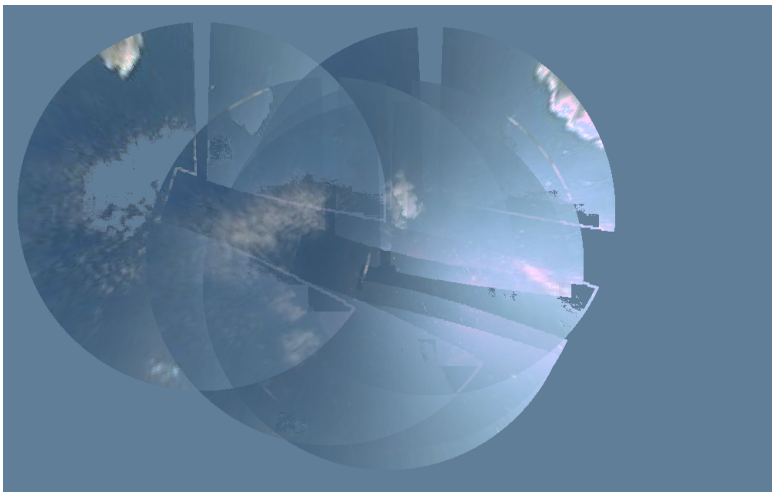
(d) Aggregation at $TSI1_t$



(e) Aggregation at $TSI2_t$

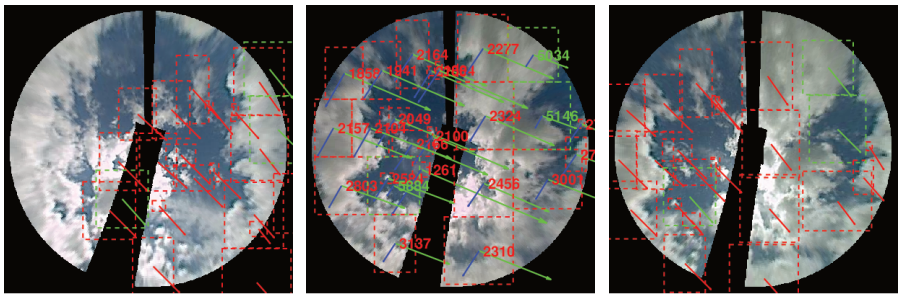


(f) Aggregation at $TSI3_t$



(g) Two-layer stitched view

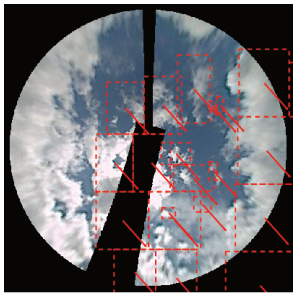
Figure 11: Determination of two cloud layers corresponding to the images shown in Figure 10. (a), (b), and (c) show the matching results for ($\mathbf{d}_m^{1,2,t}$,$\mathbf{v}_m^t$). The estimated heights are labeled in (b). (d), (e), and (f) are marked with blocks indicating the two wind fields (red and green). The arrows in the images from TSI1 and TSI3 represent motions, whereas the arrows in the images corresponding to TSI2 represent the $TSI_2 \rightarrow TSI_1$ (green) and $TSI_2 \rightarrow TSI_3$ (blue) displacement vectors.



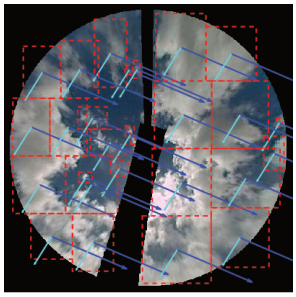
(a) Max. φ at $TSI1_t$

(b) Max. φ at $TSI2_t$

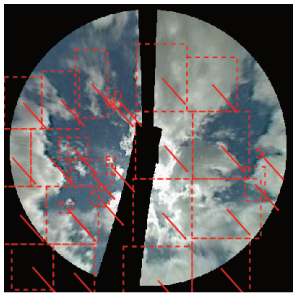
(c) Max. φ at $TSI3_t$



(d) Single WF at $TSI1_t$



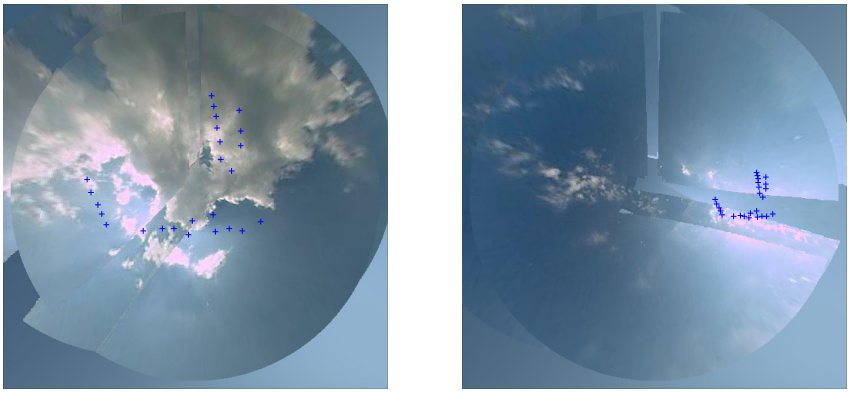
(e) Single WF at $TSI2_t$



(f) Single WF at $TSI3_t$

(g) Single-layer stitched view based on $TSI2_t$

Figure 12: The single layer detected from the nine images shown in Figure 8. (a), (b), and (c) show the tracking results for ($\mathbf{d}_m^{1,2,t}$,$\mathbf{v}_m^t$). The regions marked with green boxes in (b) display obvious bias due to the boundary effect. (d), (e), and (f) show the single-layer field, (h^t,v_x^t,v_x^t) = (2130 m, 8 px, 9 px) marked in red.



(a) h=1000 at 2:00PM (b) h=3000 at 9:00AM

Figure 13: Examples of the locations of the sun-blocking pixels (blue crosses) for the 25 pyranometers in the stitched multi-TSI view.

Table 2: Descriptions of four subsets with various cloud and weather conditions. $image_{\#}$: number of TSI images, $k_{\#}$: number of GHI measurements, cf: estimated range of cloud fractions in the sky images, cf_{σ}: mean and standard deviation of the cloud fraction, k: range of the clear-sky index. k_{σ}: the mean and standard deviation of k, exp: observed (ab)normal condition of the TSI images, CBH_{est}: cloud height range, v_{est}: cloud motion in image, $WF_{\#}$: number of cloud layers, $zenith$: solar zenith range, $condition$: cloud conditions.

Subset	D_s	D_m	D_o	D_{mix}
$image_{\#}$	2517x3	2520x3	2406x3	2520x3
$k_{\#}$	2517x25	2520x25	2406x25	2520x25
cf	[0.07,0.94]	[0.03,0.94]	[0.94,0.95]	[0,0.95]
cf_{σ}	0.71 ± 0.25	0.70 ± 0.27	0.94 ± 0.01	0.50 ± 0.35
k	[0.11,1]	[0.06,1]	[0.17,1]	[0.11,1]
k_{σ}	$0.46. \pm 0.22$	0.57 ± 0.27	0.63 ± 0.21	0.81 ± 0.23
exp	normal	underexposed	normal	green-dominant
$WF_{\#}$	1	2	2	2
$CBH_{est}(m)$	[1590,2960]	[1890, 4420]	[6020,15730]	[440 12330]
$v_{est}(px/min)$	[36,60]	[6,36]	[0,54]	[0,60]
zenith	$[40°, 57°]$	$[42°, 58°]$	$[41°, 56°]$	$[41°, 57°]$
condition	low,scattered	multi-layer	overcast	mixture

Table 3: MAE and RMSE metrics for 1-minute and 5-minute irradiance forecasts. The subscript of each score indicates the subset type used to train and test the model. The subscript "avg" indicates the average performance on the entire dataset.

Dataset	R_{shift}		$linear_{\delta}$		$linear_{all}$		SVR_{linear}		SVR_{RBF}	
	1 m	5 m	1 m	5 m	1 m	5 m	1 m	5 m	1 m	5 m
MAE_s	0.16	0.20	0.14	0.20	0.11	0.16	0.10	0.16	0.09	0.14
MAE_m	0.16	0.19	0.14	0.19	0.11	0.15	0.10	0.14	0.08	0.13
MAE_o	0.04	0.08	0.04	0.08	0.03	0.07	0.03	0.07	0.03	0.06
MAE_{mix}	0.11	0.16	0.12	0.17	0.09	0.13	0.07	0.12	0.06	0.09
$\mathbf{MAE_{avg}}$	0.12	0.17	0.12	0.17	0.10	0.16	0.09	0.15	0.07	0.12
$RMSE_s$	0.23	0.27	0.20	0.26	0.15	0.20	0.15	0.20	0.14	0.18
$RMSE_m$	0.26	0.29	0.21	0.28	0.15	0.20	0.15	0.21	0.14	0.20
$RMSE_o$	0.06	0.11	0.06	0.11	0.05	0.09	0.05	0.09	0.04	0.08
$RMSE_{mix}$	0.23	0.29	0.22	0.26	0.15	0.20	0.17	0.21	0.13	0.18
$\mathbf{RMSE_{avg}}$	0.23	0.27	0.21	0.26	0.15	0.21	0.17	0.22	0.13	0.19

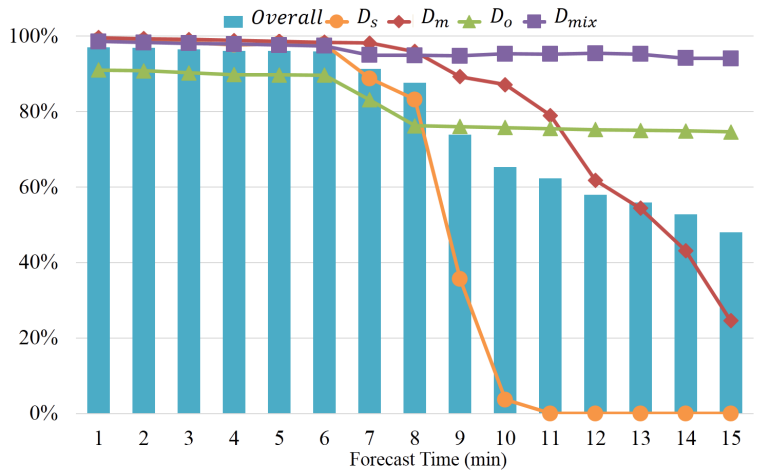
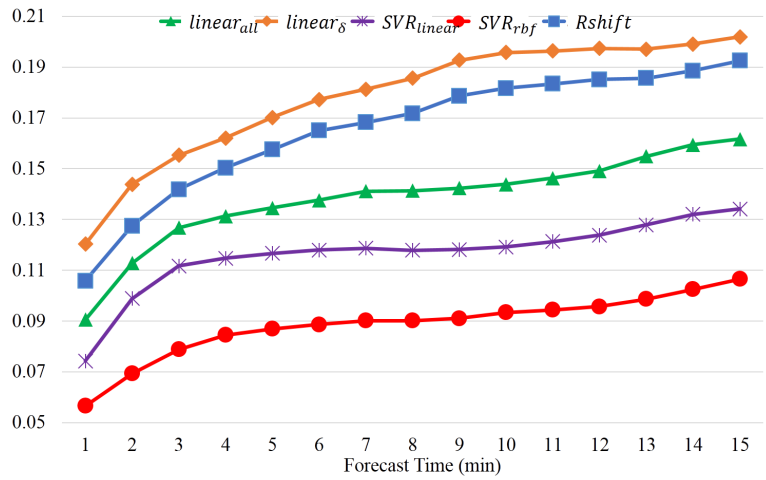


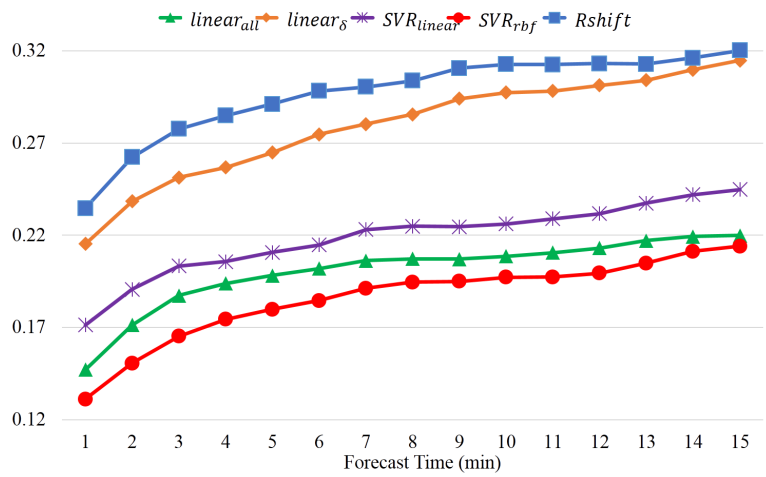
Figure 14: Successful Tracking Index (STI) values for the datasets in the 1-15 minute forecasting range. *Overall* represents the results for the entire dataset, which contains all four independent subsets.

Table 4: MAE and RMSE metrics for 10-minute and 15-minute irradiance forecasts. '-' indicates that no forecast output is available due to cloud tracking failure. '*' indicates an incomplete dataset that has a low STI value or an average performance influenced by incomplete/empty subsets.

Dataset	R_{shift}		$linear_{\delta}$		$linear_{all}$		SVR_{linear}		SVR_{RBF}	
	10 m	15 m	10 m	15 m	10 m	15 m	10 m	15 m	10 m	15 m
MAE_s	0.27*	-	0.25*	-	0.24*	-	0.16*	-	0.13*	-
MAE_m	0.20	0.21*	0.20	0.21*	0.16	0.18*	0.15	0.17*	0.13	0.15*
MAE_o	0.09	0.11	0.09	0.11	0.08	0.10	0.08	0.09	0.07	0.08
MAE_{mix}	0.18	0.19	0.20	0.20	0.14	0.16	0.12	0.13	0.09	0.11
$\mathbf{MAE_{avg}}$	0.17*	0.17*	0.18*	0.17*	0.17*	0.16*	0.15*	0.15*	0.12*	0.11*
$RMSE_s$	0.33*	-	0.30*	-	0.36*	-	0.21*	-	0.18*	-
$RMSE_m$	0.30	0.30*	0.29	0.30*	0.21	0.23*	0.22	0.24*	0.21	0.23*
$RMSE_o$	0.12	0.15	0.12	0.15	0.11	0.12	0.11	0.13	0.10	0.11
$RMSE_{mix}$	0.31	0.32	0.30	0.31	0.21	0.22	0.23	0.24	0.20	0.21
$\mathbf{RMSE_{avg}}$	0.29*	0.29*	0.28*	0.28*	0.22*	0.22*	0.23*	0.23*	0.21*	0.20*



(a) Average MAE scores for five folds



(b) Average RMSE scores for five folds

Figure 15: MAE and RMSE scores for irradiance predictions on the data subset D_{mix} over a time range from one to fifteen minutes.

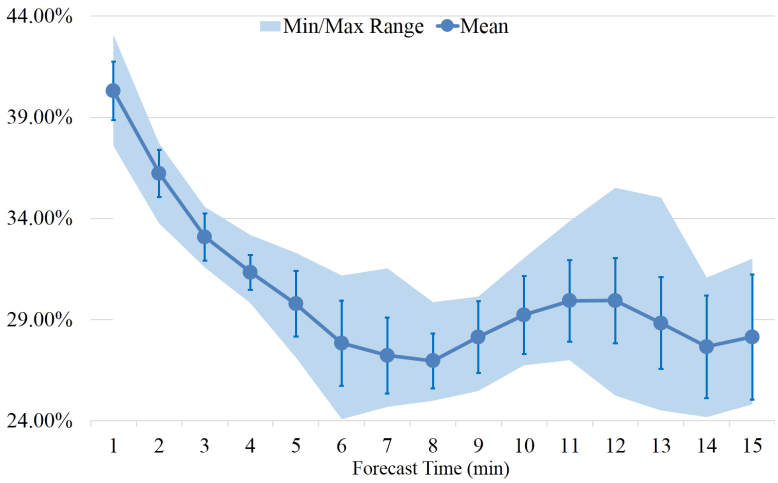
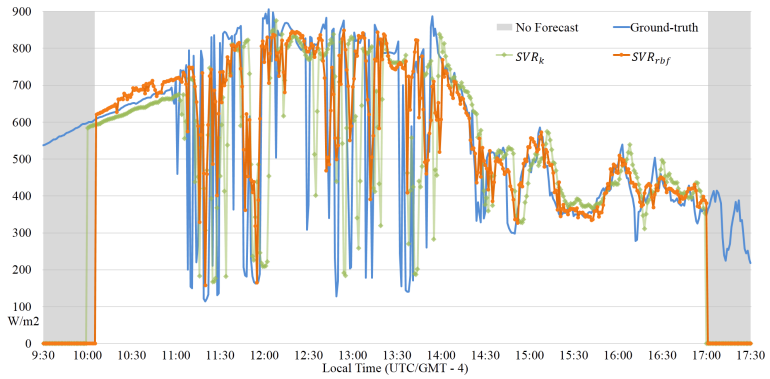
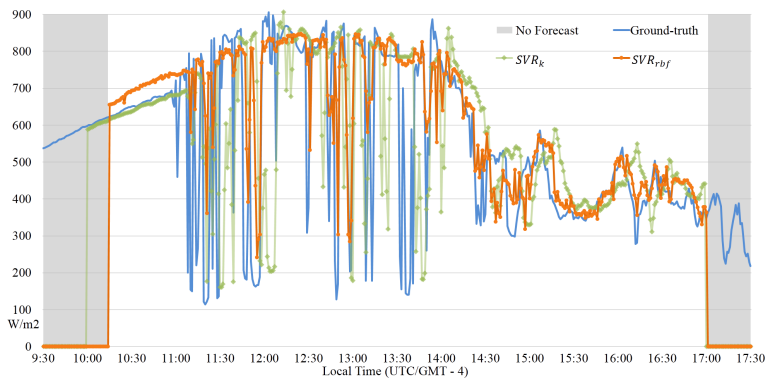


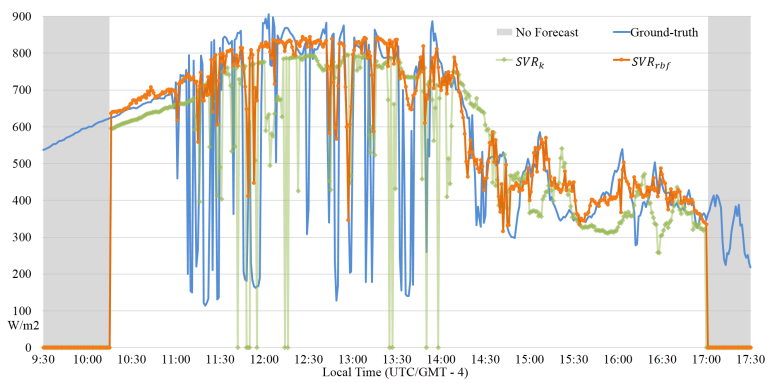
Figure 16: Improvements in the MAE ratio achieved by the non-linear SVR_{rbf} model in comparison with the persistent model on all available data. The Min/Max bounds represent the range of the percentage improvement values for all 25 stations. The average performance is denoted by the plotted line, which includes standard deviation bars on either side.



(a) 5-minute forecasts with $RMSE_{rbf} = 145.3$ and $RMSE_k = 213.5$.



(b) 10-minute forecasts with $RMSE_{rbf} = 171.3$ and $RMSE_k = 223.6$.



(c) 15-minute forecasts with $RMSE_{rbf} = 177.5$ and $RMSE_k = 241.0$.

Figure 17: Real forecasts based on our new prediction system using SVR_{rbf} and SVR_k. Gray/dark areas with a flat "0" or no forecast value represent data points that are out-of-FOV or correspond to a low zenith angle. $RMSE_{rbf}$ and $RMSE_k$ are the root-mean-square errors of SVR_{rbf} and SVR_k compared with the real GHI values.