RAS Techniques and Instruments

RASTI **4,** 1–11 (2025) https://doi.org/10.1093/rasti/rzaf034
Advance Access publication 2025 August 5

Nighttime cloud detection, tracking and prediction with All-Sky cameras

Sebastian Buntin ,★ Chris M. Copperwheat and Helen E. Jermak
Astrophysics Research Institute, Liverpool John Moores University, IC2, Liverpool Science Park, 146 Brownlow Hill, Liverpool L3 5RF, UK

Accepted 2025 July 22. Received 2025 July 5; in original form 2024 October 18

ABSTRACT

This paper presents a novel method for real-time nighttime cloud detection, tracking, and prediction using all-sky cameras, aimed at enhancing the efficiency of ground-based robotic telescopes. Ground-based telescopes are vulnerable to adverse weather conditions, particularly cloud cover, which can lead to the loss of valuable observation time and potential damage to the telescope. Existing methods for cloud detection have limitations in accuracy, particularly under varying illumination conditions such as gibbous moon phases. To address these challenges, we developed an algorithm that uses the temporal incoherence of image sequences from all-sky cameras. The method computes difference images to highlight moving cloud structures, applies Otsu thresholding to generate binary cloud maps, and uses mathematical morphology techniques to reduce noise from bright stars and other artefacts. Segmented cloud regions are then tracked across successive frames, allowing estimation of a velocity vector and enabling short-term predictions of cloud movement. Our approach achieves reliable cloud detection and tracking, providing predictions up to 15 min into the future – a capability critical for robotic telescopes that rely on look-ahead scheduling. The system was validated against extensive historical data from the Liverpool Telescope's Skycam A and T systems, achieving a false positive rate of approximately 1 per cent and a similar false negative rate, depending on cloud thickness and speed. By improving cloud forecasting and observational scheduling, the system offers a valuable tool for enhancing the operational reliability of robotic telescopes.

Key words: Algorithms – Cloud Detection – Cloud Tracking.

1 INTRODUCTION

Ground-based telescopes are susceptible to environmental factors that can adversely affect their operational integrity and scientific output. Among these environmental factors, precipitation and elevated levels of relative humidity stand out as threats, possessing the potential to inflict harm to the optical components that are fundamental to the telescopes' function and the electronic systems that control their operations. To protect the telescope from these effects, the telescope's enclosure must be closed before the environment changes to these potentially harmful conditions. Robotic telescopes like the Liverpool Telescope (LT; Steele et al. 2004) do not have human nighttime operators who can observe the weather conditions and decide if the telescope can be operated safely or not. The decision must be made by a computer system and ensure safety at all times. This can lead to very conservative strategies where the enclosure is shut in conditions where a human operator would decide otherwise.

In situations where the sky is partially cloudy and it is still safe for the telescope to operate, the telescope might observe cloudy areas and produce unusable blank frames, wasting valuable observation time, especially if other parts of the sky could have been observed instead. So it is helpful to know before an observation whether the target is visible or covered by clouds.

The Liverpool Telescope uses a dispatch scheduler, in which the robotic control software selects a new observing group from those available when the current group is concluded. One consequence of this is that a high-science-priority group might not be chosen if it is at a low altitude and rising, since the robotic controller will prefer to wait and observe it at a lower airmass. However, if weather conditions are worsening, this may be an undesirable choice. The New Robotic Telescope (NRT; Copperwheat et al. 2015) is therefore planned to be equipped with a look-ahead scheduler, a scheduling system that plans several observations ahead. This type of scheduler must be aware not only of the current cloud situation but also of the situation in the near future. The resolution of the data from weather prediction models is usually too coarse in the spatial and temporal dimensions for a detailed cloud pattern over the observatory.

Automated cloud-detection is still an active research field. In recent years, some algorithms proposed for cloud detection, including Dev et al. (2017), make use of the so-called superpixels to detect clouds. The algorithm tries to cluster segments in the image using spatial coherence and then distinguish those segments as either cloud or sky. The algorithm performs well in dark conditions but encounters problems with overexposed areas in the image, occurring especially in the gibbous moon phases. This can result in large sections of the sky around the moon being clustered to one section and labelled as cloud, no matter the true conditions.

Another method, proposed by Adam et al. (2017), tries to solve the problem of finding cloud-covered and open parts in the sky by detecting stars in the night sky and marking regions where no or only very bright stars are detected as cloud-covered. This method works also well on dark nights but has problems in nights during gibbous lunar phases where most stars are not visible in the all-sky image

★ E-mail: s.buntin@2019.ljmu.ac.uk

due to overexposure effects. The authors also use their cloud map as one object and track its movement across the sky. Problems occur again on full moon nights. The moon itself creates a large area in the image due to the overexposure caused by fixed exposure times, lens flares, and dust on the lens. This overexposed area is moving with the apparent movement of the night sky and can cover a large part of the image. In this area, star detection is not possible.

More recent and computationally more expensive methods, as shown in Mommert (2020), use neural networks to detect clouds in all-sky images. Although neural networks are an interesting solution, the system needs a lot of accurately annotated training data to provide results comparable to the ones presented in this paper.

In a paper published by Ye et al. (2022), a self-training algorithm using superpixel segmentation and machine learning is proposed for daytime sky images. The algorithm is able to train itself with a few incompletely annotated cloud images, thus reducing the training and annotation amount significantly but relies on the colour difference in the image and is thus not applicable to night-time images.

Cloud tracking is also an active field of research with many applications in the fields of weather forecasting, the energy sector (for estimating the energy produced by photovoltaic systems in smart grids), climate research, and also astronomy. The tracking is done using different methods, for example by calculating the optical flow (see Zhang et al. 2019). This approach requires a high image rate (at least 20 all-sky images per minute) to produce usable results and is not able to handle changes in the cloud structure well. Small changes in the cloud structure or a too large time gap between two frames will lead to physically implausible results.

Peng et al. (2015) use several all sky cameras distributed over a large area to generate a full 3D cloud model, including the cloud height. This allows very accurate detection and tracking but is not feasible at the Observatorio del Roque de Los Muchachos (ORM) as the area necessary to distribute the cameras is not available.

For a general overview, Zaher et al. (2017) conducted a comparative study of the algorithms available when writing their paper and Arrais et al. (2022) published an overview study on cloud tracking methods for short-term forecasting.

In this paper, a new method based on the temporal information stored in image-sequences taken by the all-sky camera at the Liverpool Telescope is introduced. Section 2 explains the cloud detection algorithm used to create the cloud maps. Section 3 explains how these cloud maps are then used to track clouds in the sky and predict their future location. Section 4 then describes how the algorithms were tested and explains the results of these tests. The method can be easily adopted for any all-sky camera system, independent of the cameras resolution or location as long as a constant time series of images is produced and a World-Coordinate System (WCS) can be fitted by a tool like Astrometry.net (Lang et al. 2010).

2 CLOUD DETECTION

All discussed cloud detection methods make use of one single image and apply methods for detecting clouds. All-sky cameras usually take images with a set frequency. For example, the all-sky camera at the Liverpool Telescope (Skycam A, Mawson, Steele & Smith 2013) takes images with an exposure time of 30 s once every minute. This results in a regular sampling of the night sky, adding another dimension to the data. With images taken at regular intervals, the temporal incoherence of the cloud structure in those images can be used to detect and track them. As clouds move with a relatively constant velocity in the night sky, the difference of two adjacent images (or even images taken over a span of 3 or more minutes) can be used to detect clouds.

Figure 1. Difference image obtained by subtracting the image shown in Fig. 2 from the image obtained one minute previously. The majority of the structure observed in this image is due to the motion of the cloud between the two exposures.

2.1 Description of the algorithm

Typically, from image to image, the only moving (changing) objects are the apparent movement of the night sky, the movement of the telescope structure, and the movement of the clouds.

We use OpenGL and the GPU's parallel processing capabilities to accelerate image normalization and difference image calculation.

The algorithm described here uses well-established image processing techniques in a novel way – to detect night-time clouds in monochromatic all-sky images. A lighting pre-processing step, critical for compensating for overexposures caused by moonlight and for normalizing pixel intensities across the image is applied first. Here, we use the FITS header parameters `BSCALE` and `BZERO` to linearly transform the raw pixel values to physical intensity values, following the relation: $physical\_value = BSCALE \times pixel\_value + BZERO$. The `BSCALE` parameter defines the scaling factor applied to each pixel, while `BZERO` represents an offset (bias) added after scaling. These parameters are commonly used to store integer data in FITS files while preserving floating-point precision.

We then apply a capping parameter `MAX` to limit the maximum allowed pixel intensity after scaling. This threshold is defined empirically based on typical saturation levels of the camera sensor and is used to compress the dynamic range, preventing bright regions (e.g. the Moon-disc) from dominating the subsequent processing.

This normalization allows subsequent difference imaging and thresholding to work effectively across a wide range of illumination conditions, including gibbous and full moon phases.

On the GPU, the FITS data (in our case with a resolution of 1392×1040 pixels) are loaded as 2D textures into texture memory, and a pixel shader is used to apply the normalization operation across all pixels in parallel. The shader program calculates the scaled and clipped pixel intensity for each pixel and outputs the result as a processed image ready for difference calculation.

Then, a difference image (see Fig. 1) of the current (Fig. 2) and the previous lighting-corrected all-sky images is taken. This is again done using a shader program. Here, both images (the current one and the 'past' image) are loaded as textures for a pixel shader which then calculated the absolute difference of each pixel.

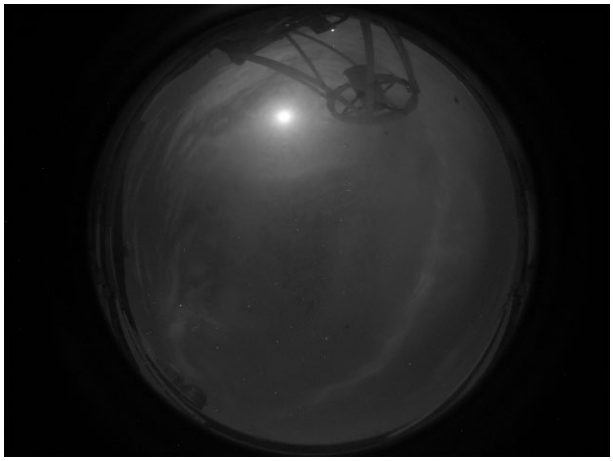

Figure 2. All-sky image taken using the Skycam a Camera at the Liverpool Telescope site at 22:45 UT on 2020 January 1. Some stars are visible as well as bands of clouds, with the cloud illuminated by the Moon at the top of the image.

While the presented implementation leverages GPU hardware for real-time difference image calculation and pre-processing, we note that the processing load (1 million pixel images at 1-min cadence) is well within the capabilities of modern multi-core CPUs. The use of GPU acceleration was motivated by our aim to scale towards higher cadence and higher resolution cameras especially in the light of the upcoming New Robotic Telescope, as well as to leverage available hardware for parallel processing. However, real-time performance for the given resolution and cadence is achievable on a CPU-based system, and GPU use is not strictly required.

2.2 OTSU binarization

The difference image is then binarized using the Otsu binarization method (see Otsu 1979). Otsu's method is particularly effective for images with bimodal histograms, where two distinct peaks represent two dominant pixel intensities, typically corresponding to the foreground and background. This method calculates a binarization threshold which determines which pixels are black or white. White pixels, in this case, represent a large change of the pixel values which is caused by cloud movement, so a white pixel is seen as a cloud pixel whilst a black pixel represents 'open sky' (no change). The moving telescope structure can either cause large differences in the pixel value and appears as cloud pixels (e.g. when slewing) or blocks the view to moving clouds having the structure appear as black pixels. As this is usually only visible in the pixels close to the horizon in areas where no observations are performed, these wrongly classified pixels do not pose a problem. As the position of the telescope is known at the time of the image analysis, the area around the structure in the cloud map can be masked and a flag can be saved alongside the masked pixels in the cloud map (see Section 2.5).

The fundamental idea of the Otsu algorithm is to automatically find the intensity threshold that separates pixels into two classes (in our case, cloud and clear sky) by maximizing the variance between these classes – that is, it selects the threshold where the difference between the average intensities of the two groups is greatest relative to their internal spread.

Otsu's method is applied to the difference image and thus separating pixels with high variability (cloud pixels) from pixels with low variability (open sky pixels).

First, an image histogram with 256 bins is created using the graphics hardware. After this step, each bin is divided by the number of total pixels, in the image (width × height) resulting in the normalized histogram (NH). The normalized histogram represents the probability of a pixel falling into a specific bin.

The next steps are then performed on the CPU for the 512-bin histogram, as these steps are not well-suited for parallel execution. Although it is possible to implement them on the GPU, the gain in speed is negligible compared to the overhead of compute shader setup and data transfer.

Let the image histogram be binned into L discrete intensity levels (typically 256 or 512 bins). We denote by NH(i) the normalized histogram value at bin i, i.e. the fraction of all pixels in the image that have intensity i. For each candidate threshold bin k, the histogram is split into two classes: C_0, containing intensity levels $[1, \ldots, k]$, and C_1, containing levels $[k+1, \ldots, L]$. The goal is to find the threshold bin k^* that maximizes the between-class variance $\sigma_B^2(k)$, providing the best separation between cloud and clear-sky pixels in the difference image.

Otsu then gives the formulae for the class probabilities:

$$\omega_0(k) = \sum_{i=1}^{k} \mathrm{NH}(i)$$

$$\omega_1(k) = \sum_{i=k+1}^{L} \mathrm{NH}(i) = 1 - \omega_0(k)$$

and the class mean intensities:

$$\mu_0(k) = \frac{\sum_{i=1}^{k} i \cdot \mathrm{NH}(i)}{\omega_0(k)}$$

$$\mu_1(k) = \frac{\sum_{i=k+1}^{L} i \cdot \mathrm{NH}(i)}{\omega_1(k)}$$

The between-class variance is then defined (see Otsu 1979) as:

$$\sigma_B^2(k) = \omega_0(k)\,\omega_1(k)\,(\mu_1(k) - \mu_0(k))^2$$

The optimal threshold k^* is the value of $k \in [1, L-1]$ that maximizes this variance:

$$k^* = \max_{1 \le k < L} \sigma_B^2(k)$$

Once the optimal threshold k^* is determined, it is applied to the difference image to produce a binary cloud map. Pixels with values greater than or equal to k^* are classified as 'cloud', and those below as 'clear sky'.

In some cases, the lighting conditions between two adjacent images can change drastically, for example, when the telescope structure moves between the camera and the moon.

To allow proper classification in these cases, an analysis was conducted over the time span from 2019 to 2023, analysing the calculated thresholds for different moon phases (lighting conditions). This resulted in a function which provides boundary conditions for the binarization threshold of two adjacent cloud maps. The calculated threshold of a new cloud map is then clamped against these boundaries based on the previous cloud map's threshold to avoid misclassification of sky and cloud pixels in the difference images.

2.3 Mathematical morphology

Stars with magnitudes below 3 mag (V band) and other bright objects like Jupiter or even the ISS can lead to artefacts in the difference image, visible as small spots covering about 5 × 5 pixels on average.

These artefacts can be removed or reduced by using techniques called Mathematical Morphology (Serra 1982) particularly erosion and dilation. The technique applies a structuring element (in this case a 3×3 pixel square, other elements such as circles or ellipses are also possible but computationally more expensive) to a binary image. The erosion is then defined mathematically as:

$$I \ominus E = \{z \in Z \mid B_z \subseteq I\},$$

where I denotes the binary image, E the structuring element (3×3 pixel square), Z an integer grid, and B_z the translation of B by the vector z.

The structuring element is applied to each pixel. For the presented algorithm it has the form of a 3×3 pixel square with

$$E = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

The structural element can be seen as a small subset of the original binarized image centred at a pixel B_z of this binary image. Then, the minimum of all pixel values of the binarized image, where the structural element has a value of 1 is calculated and written to the output image. This is repeated for each pixel in the binarized image. An example OpenGL pixel shader is given in listing 1.

```
  Listing 1: pixel shader for erosion
  #version 330 core
  in vec2 fUv; // coordinate of current pixel
  out vec4 FragColor; // output, 32bit RGB
  uniform sampler2D Tex; // input binary im-
age
  uniform float width; // width of imput im-
age
  uniform float height; // height of in-
put image

  const int frad = 1; // re-
sults in 3x3 square

  void main(void)
  {
   float w = 1.0 / width;
   float h = 1.0 / height;
   float erode = 1.0;
   for (int j = -frad; j <= frad; j++)
   {
    for (int i = -frad; i <= frad; i++)
    {
     float fi = float(i);
     float fj = float(j);
     erode = min(erode, tex-
ture2D(Tex, fUv.xy + vec2(fi * w, fj * h)).r);
    }
   }
   FragColor = vec4(erode, erode, erode, 1.0);
  }
```

Similarly to erosion, the morphological operation of dilation can be defined as $I \oplus E$. The difference is that instead of the minimum of the pixel values (resulting in erosion), the maximum is used (resulting in dilation) so that the line 21 in listing 1: `erode = min(erode, ...);` becomes `dilate = max(dilate, ...);` renaming the variable '*erode*' to '*dilate*' respectively.

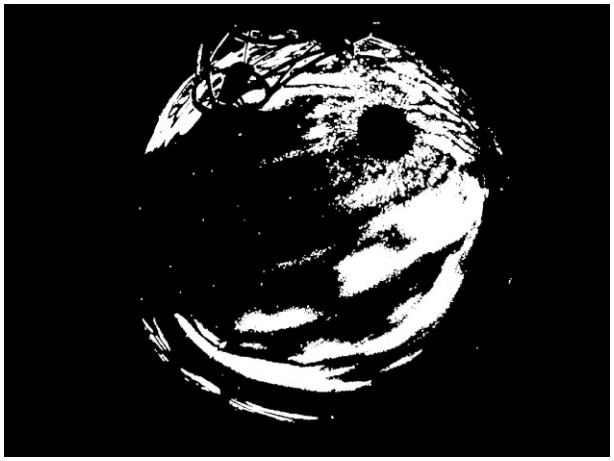

Figure 3. Binary image obtained after the application of the morphology methods (1) erosion and (2) dilation with a 3×3 filter kernel. The number of star pixels is reduced whilst the cloud structure remains intact. Larger filter kernels (e.g. 5×5 or 7×7) will remove all stars but also small cloud structures like the ones present in the middle of the image.

The result of this morphological process, applied to an unfiltered image can be seen in Fig. 3. In this image, erosion is applied to the unfiltered image and thereafter, dilation is applied to the result of the erosion step.

2.4 Further filtering

The amount of visible bright stars in the image is reduced by using the technique in Section 2.3, but some star-like structures still remain. The method can be improved for higher accuracy by applying a larger structural element (i.e. a 5×5 or 7×7 pixel square) but this will also affect the actual pictured cloud structure. To maintain the speed and keep the implementation simple and applicable to GPU hardware, a simple heuristical component detection algorithm can be applied after the erosion-dilation steps. The idea is to leverage the fast memory of modern graphics hardware and – for each pixel in the original image – analyse an area of 21×21 pixels (with the current pixel being in the centre of this area) and perform the heuristical component detection as follows: The algorithm analyses each row and then each column. If a row (or column) contains a switch from 0 to 1 (or 1 to 0) in pixel values, a marker is set stating that the row (or column) contains pixels belonging to one or more components. If a row (or column) exists between two marked rows (or columns) that only contains 0-valued (dark) pixels, it is assumed that the area contains more than one component and is discarded. If only one component is found in the area and the component is fully included in the area (i.e. not at the edge) and the area of this component is below a threshold (20 pixels), the pixels of this area are set to 0. Although there might be cases of oddly shaped areas where the algorithm cannot find a single row or column (e.g. because the areas have to be split by a diagonal), it will reduce the amount of stars visible in the cloud pattern to almost zero and provide a simple and fast approximation to the 'optimal' solution with all stars removed. This is because stars appear as small, convex shapes in elliptical form, so large groups of pixels with value 1 can be attributed to clouds.

The result can be seen in Fig. 4.

Figure 4. Binary image obtained after an additional star removal step, a simple heuristic is applied to a 21 × 21 pixel window moved over the whole image, again leveraging fast GPU hardware, to test for small star-shaped pixel areas.

2.5 Storage and query

The generated cloud map is then stored as a FITS file with the world-coordinate system from the initial all-sky image on the file system. The difference image is stored in a similar way. Meta-information for this cloud-map is stored in a SQL Data base. This includes the timestamp of the cloud image, the OTSU threshold and a parameter called the '*total cloudiness*' which is simply the number of cloud pixels over the total number of pixels in the area of interest (in the case of the LT implementation, the area of interest is considered to be all pixels within a circle around the centre of the image corresponding to an altitude of 20°).

3 CLOUD TRACKING AND PREDICTION

3.1 Individual cloud identification

While cloud detection is helpful to determine the current sky cover and check if an observation would be feasible *now*, cloud tracking tries to identify individual clouds in the image and track their movement over time. This will allow to predict their location in the near future and support *planning ahead* to help schedule observations which will be in areas of the sky where there will be no clouds within a certain time span. Any prediction will inevitably decrease in accuracy as the time span increases. Most Liverpool Telescope observation groups are relatively short (≈ 15 min, including overheads) so this time has been adopted as a baseline for making predictions. This effectively can give the robotic scheduler the ability to decide whether a group is better observed *now* or *next* and the decision can be revisited after each observation is completed.

The wind in the cloud layers of the atmosphere remains constant and does not suddenly change its direction, meaning clouds move in a constant direction for a relatively long period of time. Analysing the Liverpool Telescope Skycam A all-sky movies and conducting own experiments using an all-sky camera in Emleben (Thuringia, Germany), the shortest time for the general cloud movement direction changing by more than 60 deg was about two to three hours for wind speeds at which observations can be safely conducted. The actual speed at which the clouds are moving varies in similar time-scales. It is thus acceptable to assume a constant cloud velocity vector for a prediction time of 20 to 30 min.

In a first step, the fisheye projection of the all-sky camera has to be removed, the image has to be unprojected as the fisheye effect can be described as a projection of the full hemisphere over the camera into a circular region of the sensor. This un-projection is necessary as the clouds will not travel in straight paths through the original images but the cloud path will appear warped and bent. Also the shape of the cloud is distorted based on the position in the original image.

As the all-sky image covers a 180° field of view, pixels near the horizon correspond to very large physical areas on the ground or in the lower atmosphere (up to hundreds of square metres per pixel), while pixels near the zenith cover much smaller physical areas. This effect arises from the projection geometry, as the angular resolution per pixel (in square degrees) is roughly uniform, but the physical footprint increases dramatically towards the horizon. The de-fishing process leads to over- and under-representation in the unprojected image. To account for this, cloud identification is limited to altitudes between 20° and 90°.

To detect and identify clouds in the un-projected binary image, a connected-component labelling (CCL) algorithm using directed acyclic graphs, called Spaghetti Labelling as described in Bolelli et al. (2020), is used. Bolelli, Allegretti & Grana (2021) propose a GPU implementation for this algorithm but as the CPU implementation is already very efficient, the algorithm described in this paper does perform the CCL on the CPU.

The CCL method identifies and labels the cloud areas which form single, connected areas in the image, as well as determining their centroid and their bounding box in a binary cloud pattern image.

To speed up the following processes, identified cloud patches with an area below a set threshold (in this case 40 pixels) are discarded. In a future version it is planned to test if these small patches can be merged with nearby larger patches.

The information (label number, centroid coordinates, bounding box, and area size) for each individual cloud patch is then stored. This process is repeated for every cloud map in a sequence.

One problem arising was that parts of the moving telescopes and the moon (if visible) were also detected as cloud-like structures and marked in the binary image due to the nature of the cloud detection algorithm (see Section 2). As this is usually not a problem for cloud lookups to detect clouds at a specific moment, it might actually lead to false results further in the cloud tracking, e.g. if a part of the moving telescope (see Fig. 5) is labelled as cloud and moving to the opposite direction as the clouds. The position of the moon and its phase can be determined using well-known astronomical algorithms as described in Meeus (1998). To keep the box of the telescope small, the current azimuth and altitude of the telescope must be known to the algorithm to shrink the affected area to a minimum. This is currently done using the world-coordinate system of images recorded by Skycam T, a piggy-back small telescope attached to the structure of the Liverpool Telescope (see Mawson et al. 2013). Cloud centroids which are within the moon- or telescope area will be discarded as well.

Fig. 6 shows a cloud pattern (dark red) with the detected cloud patches and their bounding boxes (yellow).

3.2 Cloud tracking

The detected cloud patches are then matched to clouds detected in the previous cloud pattern images using a metric d based on the euclidean distance of the centroids and the overlapping area of the bounding box ($c_{x,y}$ representing the centroid position in pixel coordinates and

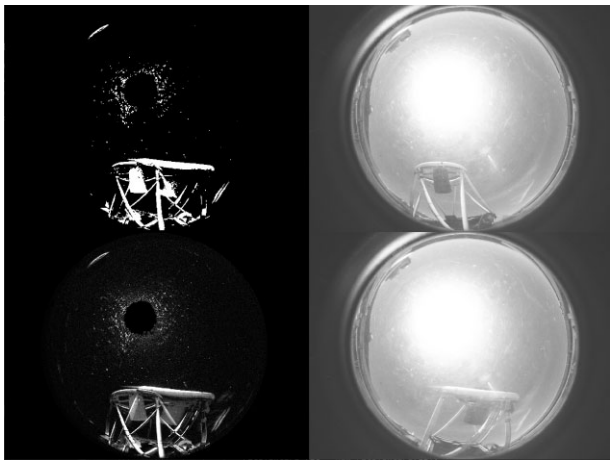

Figure 5. The fast slewing of the telescope and the full moon on the telescope structure caused the algorithm to detect the structure as a quite large cloud. The resulting cloud map can not be used for large parts of the sky. To detect this issue, large, sudden appearances of 'cloud-like structures' will be discarded.

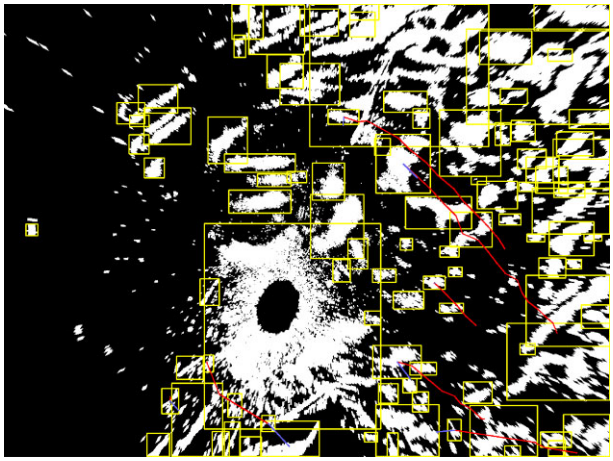

Figure 6. Detected cloud-like structures with their tracks in a cloud-map with the fish-eye effect removed. The yellow boxes mark the detected cloud-like structures in the image and the associated track for each detected cloud (the position of the centroid in the previous images) is displayed. It is also visible that the structure caused by dust and the over-exposure around the full Moon is clearly visible and detected as a cloud-like structure. As the resulting path of this structure differs from the other cloud paths, it is discarded.

A_n the area of the cloud in pixels).

$$d = \sqrt{(c_{x_2} - c_{x_1})^2 + (c_{y_2} - c_{y_1})^2} + \frac{A_1 + A_2}{\text{Overlap}(A_1, A_2)}. \quad (1)$$

This is done by evaluating the metric for each cloud patch in the current image against all cloud patches of the previous image. A 'match' between two clouds is found (e.g. the two cloud patches represent the same cloud in two adjacent images) for the patches resulting in the smallest value of d. To avoid errors d is also checked against a threshold. If d is larger than this threshold, the match is discarded as the distance between the cloud patches is too large and thus cloud patches are considered independent. Tests of various thresholds have shown that a threshold of 200 is a good estimate for the cloud system at the Liverpool Telescope.

The coordinates of the centroids for each matching cloud component are then added to a data structure called 'track'. If no matching track is found, a new track is created. This data structure contains the coordinates of the centroids of all matched cloud patches in chronological order.

The track itself is then formed of the path described by concatenating the vectors from centroid point n to centroid point $n + 1$, starting at point 1.

All tracks are then analysed and a metric for the 'smoothness' of the track is determined. This smoothness is based on the angle between two adjacent vectors (which must not be larger than a threshold, here 10 deg is used). This can be efficiently calculated using the inner product.

In this smoothing step, tracks which contain angles above the threshold are not considered 'cloud paths' and ignored. The remaining tracks are then 'smoothed' using a Gaussian filter (see Shapiro & Stockman 2001).

3.3 Cloud position prediction

Each track is then fed into a separate Kalman filter (see Kalman 1960) to predict its future direction. A Kalman filter – also known as linear quadratic estimation – is an algorithm for estimating unknown variables using a series of measurements observed over time, including statistical noise and other inaccuracies.

The predicted direction vectors produced by the Kalman filter for each cloud track are averaged to estimate the overall cloud motion, resulting in a mean velocity vector $\vec{v}$. Typically, 5 to 10 consecutive cloud patterns are sufficient to make a reliable prediction. To ensure robustness, the averaging process includes a two-pass approach: first, the general direction of motion is determined, and secondly, the magnitude of $\vec{v}$ is refined by weighting vectors pointing within $\pm 10^\circ$ of the estimated direction more heavily. The final displacement vector $\vec{v}$ is then applied to each cloud pixel in the unprojected all-sky cloud map to predict its location n minutes into the future. This calculation can be efficiently performed on the GPU using a shader program.

A GPU pixel shader is then used to shift the detected cloud blobs by that vector $\vec{v}$ to determine the cloud map at time $t + 1$. This is repeated for each minute in the future until the maximum prediction time (usually 20 min) is reached.

A problem occurring is that the amount and location of clouds coming from the horizon is unknown. Due to the resolution of the camera and the angle, only a few pixels of the camera cover a large area of the sky (undersampling) and thus reliable cloud information cannot be retrieved.

4 RESULTS AND DISCUSSIONS

4.1 Cloud detection

In order to test the efficiency of this cloud detection algorithm, it was tested against two samples in the historical data set of Skycam-A-images. The first sample was obtained from 2019 February 20 to 2022 December 31 and the second sample was obtained from 2024 January 1 to 30. The first sample consisted of 619 421 images and the second sample consisted of 16 937 images. Over these periods a variety of degrees of cloud cover were observed, from nights where conditions were fully photometric to nights where no stars were visible at all. The second sample was chosen as due to changes of our satellite image provider, in that time span the Liverpool Telescope operated without an external cloud sensor and thus was operating in cloud conditions that normally would have been resulted in observations being suspended.

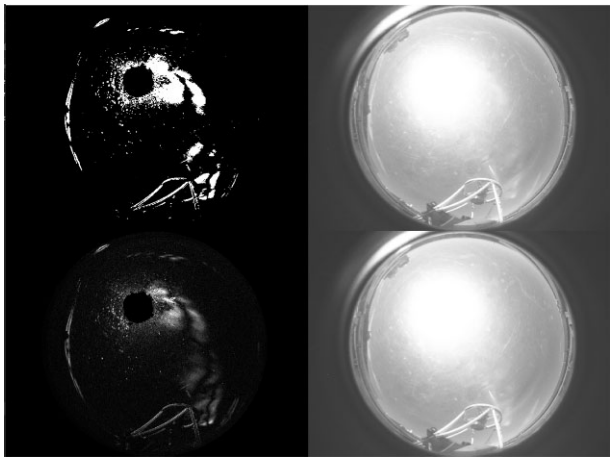

Figure 7. Principle work of the algorithm: The right column shows the input images at t and $t + 1$ min, the resulting difference image is shown on the bottom left and finally, the binarized cloud map is shown in the top left.

The algorithm was tested against Adam et al. (2017) (see Section 1) who generously provided their code on Github (https://github.com/tudo-astroparticlephysics/starry_night) as well as a method using the Skycam T at the Liverpool Telescope.

The comparison of Adam et al. (2017) against the algorithm presented in this paper, both using the Skycam A, could only be performed in dark nights when there was no visible moon. During times with visible moon, especially with a moon disc of 30 per cent or more, their algorithm was unable to perform photometry in large areas of the sky. With the full moon up, the method did not provide any useful results at all. This is caused by hardware limitations of the all-sky camera equipment at the Liverpool Telescope (Starlight Xpress Oculus). However, the problem of overexposed areas around the visible moon is a common problem on most typical all-sky cameras.

In dark nights both algorithms matched against each other by around 92 per cent of detected cloud area, depending on the density of catalogue stars in the area of the sky. A limitation of their algorithm is the fact that it can only outline clouds based on visible and non-visible stars in a provided catalogue whilst the method presented in this paper is able to accurately detect the shape of the cloud which is the main cause for the average match of 'only' 92 per cent.

In bright nights (moon disc $>$ 30 per cent), whilst their algorithm failed to provide a cloud map (using the images recorded with the equipment at the Liverpool Telescope), the method presented in this paper was able to accurately detect clouds around the actual moon disc. In Fig. 7, the right side represents two all-sky images taken one minute apart. The bottom left side is the difference image and the top left picture contains the extracted cloud pattern. The cloud pattern contains various small white patches falsely identified as clouds. These are the result of moonlight scattering on dust accumulated on the camera protection over time. With an exposure time of 30s, the moon disc itself saturates the CCD completely and the ADU values in the pixel regions around the moon disc are also close to saturation levels resulting in visible noise. This noise is falsely labelled as a cloud but it has little to no influence on the observations itself as observations this close to the moon are very rare.

The next test was conducted with the help of our Skycam T. Skycam T is a Starlight Xpress Trius SX-35 camera mounted on the top-end of the Liverpool Telescope and parallel points with the telescope. The camera currently uses a Zeiss Planar T 85 mm f/1.4 ZF2 lens which results in a pixel scale of about 44 arcsec and a field-of-view of roughly 24 $\times$ 16° and a detection limit of about 12 mag. Skycam T takes a 10 s exposure every minute and stores it in a public available data base on the website of the Liverpool Telescope. All Skycam T images are plate-solved using the Astromentry.net software suite (Lang et al. 2010) and a world coordinate system (WCS) is added to the FITS file.

To use this system to test the cloud detection system, a photometric analysis of the Skycam T images using the SExtractor tool (Bertin & Arnouts 1996) was performed. The images of cloud-free nights were analysed with the APASS catalogue (Henden et al. 2018). APASS stars are extracted from the images to calibrate the instrument magnitude.

In the next step, nights with clouds were randomly selected from the data base. Then, for every Skycam T image in these nights, the sky location of the centre pixel (in terms of Right Ascension and Declination) was determined and projected into the cloud map image and an area of about 15° around this pixel is then read from the cloud map and the 'cloudiness' parameter is calculated as described in Section 2.5.

For the all-sky camera, we determine which pixels correspond to the Skycam-T field of view. We then calculate the 'cloudiness' parameter, which is the fraction of these pixels which are classified as cloud using our detection algorithm. For Skycam T, the APASS data base is queried for the same section of the sky, and all stars with magnitudes smaller than 10 mag are retrieved. The number of stars in the data base query result determines the stars that *should* be visible (expected number of stars), which is then compared to the photometric analysis performed with SExtractor, determining the stars that actually *are* visible. We consider a region in the all-sky cloud map to be cloud-covered if more than 75 per cent of the pixels in that region are classified as cloud. This threshold mirrors the classification approach used for Skycam T, allowing for direct comparison between both systems.

If the number of visible stars is below 75 per cent of the expected number of stars, the image is labelled as 'cloud covered'. The threshold of 75 per cent was chosen due to the optical limitations of Skycam T. As it is a medium-field camera with a relatively large field-of-view and SExtractor only performs aperture photometry, especially in areas with a high star density (crowded fields) SExtractor might not be able to extract all APASS stars without using more sophisticated techniques like PSF photometry. To account for thin clouds and to calculate the detection threshold of the cloud system, a second analysis is performed if the detected number of starts is above the 75 per cent threshold of the expected stars. Here, the measured star magnitudes are compared against the catalogue values (using the calibration performed on a clear night) and if the average magnitude is below 3 mag of the catalogue value, the image is also labelled as 'cloud covered'. Clouds which dim the bright APASS stars with less than 10 mag by at least 3 mag are also detected by the all-sky cloud monitor.

For the cloud detection evaluation, 47 nights from 2019 to 2023 were selected to represent a wide range of observing conditions, including bright nights with and without the Moon, as well as dark nights. These nights were identified via entries in the night log[1] and visual inspection of the nightly all-sky video sequences.[2] From a total of 32 307 all-sky images across these nights, the first and last 60 min of each night were excluded to remove twilight effects. From

[1] https://telescope.ljmu.ac.uk/Reports/
[2] https://telescope.livjm.ac.uk/data/webfiles/Skycam/browse2025.html

Table 1. Confusion matrices for cloud detection under three observing conditions. Each matrix is normalized such that values sum to 1.0. Rows correspond to **predicted** class, columns to **actual** class. C = **C**loud, O = **O**pen sky. 'Bright < 25°' refers to bright nights with the target close to the Moon (rare); 'Bright > 25°' to nights with large Moon-target separation; and 'Dark Nights' to low-illumination cases. Group sizes: 100 (Bright < 25°), 1000 (others). The bottom rows of the table display **A**ccuracy and **P**recision. All values rounded to two decimals.

Bright > 25°			Bright < 25°			Dark Nights		
	C	O		C	O		C	O
C	0.46	0.01	C	0.41	0.13	C	0.41	0.01
O	0.01	0.52	O	0.00	0.46	O	0.14	0.44
A	–	0.98	A	–	0.87	A	–	0.85
P	–	0.97	P	–	0.76	P	–	0.97

the remaining images, one 100-image set and two 1000-image sets were randomly selected:

(i) **Group 1**: 100 images from bright nights (Moon altitude > −5°, Moon disc > 10 per cent) where the angular separation between the target and the Moon was less than 25°. This configuration is relatively rare.

(ii) **Group 2**: 1000 images from bright nights with Moon-target angular separation greater than 25°.

(iii) **Group 3**: 1000 images from dark nights (Moon altitude < −5° and Moon disc < 10 per cent).

Testing the cloud detection system using the Skycam T method, the following results were obtained:

For bright nights we have a false negative rate (the sky area in the cloud map is labelled 'clear' whilst is is actually cloud covered) of less than 1 per cent of the analysed images and a false positive rate also of 1 per cent outside a radius of 25° of the moon disc. Inside this area, the false positive rate can climb up to 10 − 15 per cent (especially with a visible moon disc of more than 50 per cent) due to moonlight scattering on dirt and dust on the sensor cover as well as lens-flare effects. As astronomical observations are rarely performed so close to the moon, this relatively high false positive rate is of no concern. The rate could be improved by frequently cleaning the camera's protective elements and using a sensor with a higher dynamic range.

In dark nights, the false positive rate is similar to the bright nights of about 1 per cent. Relatively thin, large, and slow-moving clouds can impose a problem on the false negative rate. Due to technical limitations of the camera, the contrast between the cloud and the open sky is too low to impact the difference image (see Section 2). In these cases, a false negative rate up to 15 per cent of the images is possible and was observed of a total of 17 nights in the years 2020 and 2021. A confusion matrix is given in Table 1.

4.2 Cloud tracking

The cloud tracking was tested against historic data. Out of the large data set of roughly 620 000 images from Skycam A covering the years 2019 to 2022, several nights with cloud cover were selected. The test involved analysing each night, and – if cloud movement was detected – applying the tracking algorithm described in Section 3. This was done by simulating the cloud detection system in real-time: images were fed sequentially into the system, which then performed both cloud detection and tracking.

For randomly selected moments during each night, the prediction module was triggered, generating predicted cloud maps for each minute up to 30 min into the future. These predictions were then compared to the actual cloud maps at those future time points. Since such 'future' maps would not be available during live operations, the prediction was also compared against the latest available cloud map (i.e. the starting point of the prediction series), to test whether the prediction provided a measurable improvement.

We selected nights that reflect realistic mixed conditions – clear nights and nights with intermittent or partial cloud cover. The cloud tracking algorithm is not relevant during long overcast periods (in which satellite forecasts suffice and observations are suspended), so the chosen nights represent scenarios where real-time prediction could inform scheduling decisions.

To compare the predicted and actual cloud maps, we extracted a circular region with a radius of 25 pixels around a representative target in each image. In each region, we calculated the 'cloudiness' parameter, defined as the ratio of cloud-pixels to total pixels. These values were compared for the predicted, actual, and baseline (non-predicted) maps. Fig. 8 shows six representative sequences.

These six sequences shown in Fig. 8 were selected to illustrate a range of cloud behaviours and prediction outcomes. They are not chosen for peak performance but rather to reflect typical scenarios the system encounters. We have excluded rare edge cases where prediction accuracy extended beyond 30 min, as these were atypical and not statistically representative.

The main limiting factor is the distortion near the horizon due to the fisheye projection. Pixels close to the horizon correspond to large, distorted regions of the sky, and incoming clouds in these areas cannot be reliably predicted. The effective forecast horizon is therefore strongly influenced by wind speed and direction: slower moving clouds can be tracked longer, while fast-approaching clouds from off-frame regions limit the prediction range.

The expected scheduling strategy for the New Robotic Telescope (NRT) is a look-ahead approach, with a planning horizon of approximately 15–20 min. As NRT will be required to respond to transient alerts (e.g. from the Vera Rubin Observatory, ZTF, LIGO, CTA), this prediction window aligns well with the typical duration of high-priority observations (usually around 10 min). Reliable short-term sky forecasts can thus play a valuable role in optimizing telescope scheduling, particularly under dynamic or marginal weather conditions.

As can be seen in the data (see Fig. 8), a prediction up to 30 min is usable. During the analysis of the historic data of the Liverpool Telescope it became clear that most groups – once picked by the scheduler and started by the Robotic Control System (see Fraser & Steele 2004) – are finished after 15 to 20 min matching well with the prediction horizon of the tracking and prediction algorithm presented in this paper.

To compare the two maps, a small circular region with radius of 25 pixels around the target was extracted from the 'prediction cloud map', the actual cloud map at the point in time the prediction represents and the cloud map at the starting point of the prediction (the latest available one in a real scenario). Then, the cloudiness parameter was calculated (the ratio of cloud pixels over total pixels) in each region and plotted in Fig. 8.

The results show that predictions up until 15 min into the future can be made with a high accuracy. After this time, the accuracy drops. Predictions for more than 30 min are unreliable. Depending on the location of the target of the sky and the wind direction, the prediction times can also be significantly shorter (e.g. if the wind is coming from the east and a target is in the eastern part of the sky, the data needed to do a prediction of 15 min and more would lie outside the image bounds).

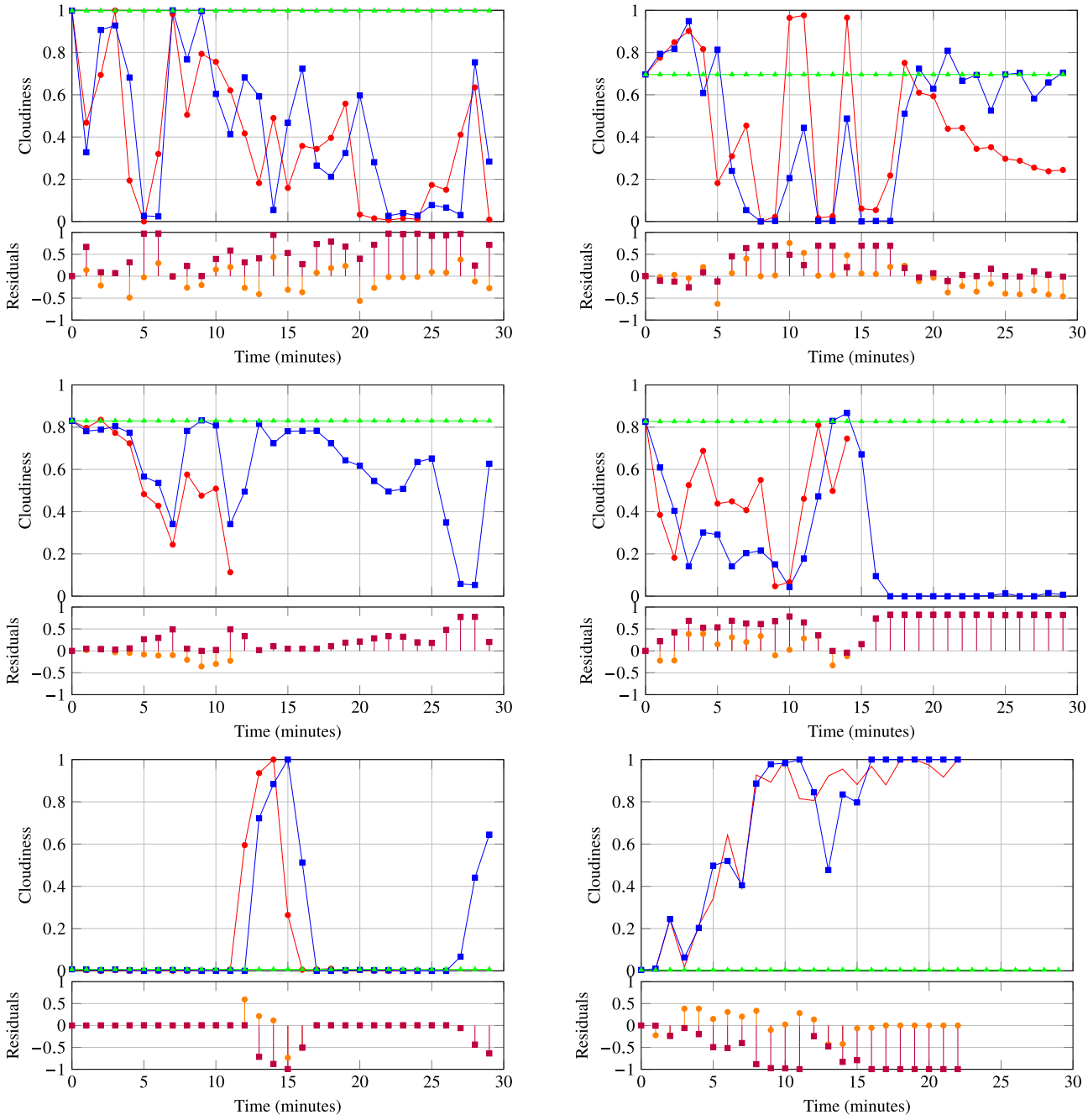

Figure 8. The plots show the accuracy for six different predictions: Plot 1 (top left) starting at 03:27 on 2020 April 9, Plot 2 (top right) starting at 02:22 on 2020 April 8, Plot 3 (middle left) starting at 20:15 on 2023 November 25, Plot 4 (middle right) starting at 22:50 on 2021 April 26, Plot 5 (bottom left) starting at 23:51 on 2020 December 31, Plot 6 (bottom right) starting at 06:16 on 2020 October 18. The graphs in the top-part show the *cloudiness* in per cent for the *next x minutes*. The cloudiness is a measure for the amount of cloud-pixels in the area of a scheduled observation (a circular region with a radius of 20 pixels around the target) over the total amount of pixels in this area. The green graph (the so-called baseline) represents the cloudiness obtained using the cloud map from the last all-sky image as a reference for the future. The red graph (prediction) shows the cloudiness obtained using the prediction method presented in this paper. The blue graph (ground truth) shows the actual cloudiness measured at that point in time. This is put in as a reference, as this data would not be available in a real-time scenario. The residuals show the difference between the baseline and the prediction (orange) as well as the baseline and the ground truth (red), again, put in as a reference. The predictions in plots 3, 4, and 5 do not have the full prediction data set, because the observations were made in the area of the sky where the wind was coming from, thus there is not enough cloud data for a longer prediction. Plot 5 shows passing clouds. Plot 6 ends after 22 min as the weather control system shut the telescope enclosure because of incoming severe conditions.

The main limiting factor is that only little knowledge of incoming clouds is available in the images. Due to the fish-eye effect, a pixel close to the horizon spans a large area of the sky with an extremely high distortion compared to pixels in the centre of the image. This causes the incoming clouds to be severely distorted in a way that cannot be fully accounted for in the algorithm. It is thus not possible to reconstruct the cloud pattern after a certain time. This time is closely correlated to the wind speed in the cloud layer. In low wind

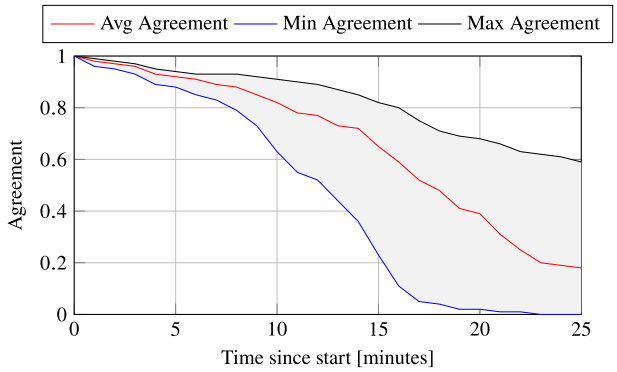

Figure 9. Agreement between predicted and actual cloud maps over time, averaged across 100 sequences. The red line shows the average agreement, while the shaded area spans the minimum and maximum values observed at each time-step. Agreement is defined as the fraction of correctly matched cloud pixels in the region of interest.

speeds the clouds move slower and predictions can be made up to 20 min more. Two cases were found (but not included in the data used for the graph in Fig. 8) were predictions up to 35 min still provided usable results, but as these were rare, hand-picked events, they were excluded. A study is in preparation to see if a correlation between the wind-speed at ground level and the cloud speed can be determined. For a more detailed explanation of the atmospheric properties of La Palma, Varela et al. (2008) provides a good overview.

To provide a statistically robust evaluation of the cloud tracking and prediction algorithm, we analysed 100 cloud sequences extracted from historical Skycam A data between 2020 and 2022. Nights were pre-filtered to exclude fully clear or fully overcast conditions using a threshold on global cloudiness, and from the remaining mixed-condition nights, sequences of at least 30 min with visible cloud motion were randomly selected. For each sequence, the system generated predicted cloud maps up to 25 min into the future. These were compared against the actual cloud maps at each time-step using a circular region of interest (ROI) with 25-pixel radius, randomly placed within the altitude range of 35° to 80° to minimize distortion effects near the horizon and zenith. The agreement metric, defined as the fraction of matching cloud pixels in the ROI, was used to quantify prediction accuracy. The results, shown in Fig. 9, present average, minimum, and maximum agreement values across all 100 sequences. Agreement remains high during the first 10–15 min and decreases gradually beyond that, depending on cloud structure and dynamics. A second, independently selected test set confirmed that 100 sequences are sufficient to yield stable statistics (agreement deviation <3 per cent). The six example sequences in Fig. 8 were included to illustrate *typical* visual behaviour and were not selected based on performance quality.

All evaluation was done using the all-sky camera sequence itself as ground truth. We did not use Skycam T for this purpose because its small and variable field of view, dictated by the telescope's pointing, prevents consistent tracking of cloud structures over time.

The planned scheduling strategy for the New Robotic Telescope is a look-ahead scheduling strategy with a time horizon of about 15 to 20 min into the future. As the NRT will be used to react to alerts from optical surveys as the Vera Rubin Observatory or the Zwicky Transient Facility as well as other surveys like the LIGO Gravitational Wave Observatory or the Cherenkov Telescope Array. The NRT will thus have to process many different targets with short exposures (around 10 min of exposure time) so a prediction of the sky conditions over the next 10 to 30 min will be a useful contribution to the scheduling decision, especially in mixed conditions.

5 CONCLUSION

In this paper a novel set of algorithms for cloud detection, tracking, and the prediction of their future location were presented. The cloud detection algorithm uses difference images taken on regular intervals with an all-sky camera and uses binarization techniques to detect clouds in these all-sky images.

The algorithm is fast and – with the use of libraries as Astrometry.net (Lang et al. 2010) – does only need little calibration, depending on the exposure settings of the all-sky camera. Re-calibration is only necessary if the camera hardware itself or the orientation is changed. It can be plugged into the data feed of an all-sky camera and provide cloud patterns for Right Ascension and Declination coordinates provided to a Web-API. The system should ideally be positioned so that moving telescope parts are not visible (as it is the case at the LT) to avoid either masking a rather large area or misclassify the moving structure as cloud.

The robustness of the algorithm was tested against (Adam et al. 2017) and provided comparable results in dark nights and outperformed the other system in bright nights.

The main limiting factor of this algorithm is the binary decision if a pixel represents either 'cloud' or 'clear sky' and is not able to determine the opacity of the cloud e.g. how much fainter a star would be through a thin cloud. A future version will implement the photometric approach of Adam et al. (2017) in combination with the presented cloud detection by measuring visible stars in the area of a cloud pattern.

The cloud tracking and detection system sits on top of the cloud detection system and analyses a time-series of clouds to detect individual clouds and track their movement through the image plane to determine a common tracking vector. This tracking vector is then used to predict future positions of the cloud pixels and generated predicted cloud patterns. Up to 15 min in the future (with longer prediction times possible with lower wind speeds in the cloud layer), predictions with acceptable accuracy can be made. This is a sufficient amount of time for the look-ahead scheduler planned for the New Robotic Telescope.

The system makes use of widely available hardware to provide real-time results of the cloud patterns and the predictions and provides a calibration-free setup as well as a Web-API for easy access to the data.

The results showed that clouds can be detected with a very high precision with a false positive rate of around 1 per cent (in areas with an angular distance > 30 per cent of the moon) and a false negative rate of 1 per cent, except in dark nights with slow-moving clouds. The cloud prediction can be performed with an accuracy average of 65 per cent for a prediction of 15 min into the future.

DATA AVAILABILITY

All data used for this paper is available as FITS files at https://telescope.livjm.ac.uk in the Skycam Archive. The source code is available on request from the authors and will be made public once the underlying project is finished.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

ACKNOWLEDGEMENTS

The Liverpool Telescope is operated on the island of La Palma by Liverpool John Moores University in the Spanish Observatorio del Roque de los Muchachos of the Instituto de Astrofisica de Canarias with financial support from the UK Science and Technology Facilities Council. The telescope and Chris M. Copperwheat receive funding from UKRI grant number ST/X005933/1. Chris M. Copperwheat and Helen E. Jermak receive funding from UKRI grant number ST/W001934/1. We thank the anonymous referee for their useful comments which led to some significant improvements to this paper.

REFERENCES

Adam J., Buss J., Brügge K., Nöthe M., Rhode W., 2017, EPJ Web Conf., 144, 01004
Arrais J. M., Martins B. J., Chaves T. Z. L., Cerentini A., Netto S. L. M., Von Wangenheim A., 2022
Bertin E., Arnouts S., 1996, A&AS, 117, 393
Bolelli F., Allegretti S., Baraldi L., Grana C., 2020, IEEE Trans. Image Process., 29, 1999
Bolelli F., Allegretti S., Grana C., 2021, IEEE Trans. Pattern Anal. Mach. Intell., 44, 1
Copperwheat C. M. et al., 2015, Exp. Astron., 39, 119
Dev S., Savoy F. M., Lee Y. H., Winkler S., 2017, IEEE International Conference on Image Processing (ICIP). IEEE, p. 345
Fraser S., Steele I. A., 2004, in Quinn P. J., Bridger A., eds, Proc. SPIE Conf. Ser. Vol. 5493, Optimizing Scientific Return for Astronomy through Information Technologies. SPIE, Bellingham, p. 331
Henden A. A., Levine S., Terrell D., Welch D. L., Munari U., Kloppenborg B. K., 2018, in American Astronomical Society Meeting Abstracts #232, p. 223.06
Kalman R. E., 1960, J. Basic Eng., 82, 35
Lang D., Hogg D. W., Mierle K., Blanton M., Roweis S., 2010, AJ, 139, 1782
Mawson N. R., Steele I. A., Smith R. J., 2013, Astron. Nachr., 334, 729
Meeus J., 1998, Astronomical Algorithms, 2nd edn. Willmann-Bell, Richmond, VA
Mommert M., 2020, AJ, 159, 178
Otsu N., 1979, IEEE Trans. Syst. Man Cyber., 9, 62
Peng Z., Yu D., Huang D., Heiser J., Yoo S., Kalb P., 2015, Sol. Energy, 118, 496
Serra J. P., 1982, Image Analysis and Mathematical Morphology. Academic Press, London, New York
Shapiro L. G., Stockman G. C., 2001, Computer Vision. Prentice Hall, Upper Saddle River, NJ
Steele I. A. et al., 2004, in Oschmann J. M., Jr, ed., Proc. SPI Conf. Ser. Vol. 5489, Ground-based Telescopes. SPIE, Bellingham, p. 679
Varela A. M., Bertolin C., Muñoz-Tuñón C., Ortolani S., Fuensalida J. J., 2008, MNRAS, 391, 507
Ye L., Wang Y., Cao Z., Yang Z., Min H., 2022, Earth Space Sci., 9, e2022EA002220
Zaher A., Thil S., Nou J., Traoré A., Grieu S., 2017, IFAC-PapersOnLine, 50, 5934
Zhang S., Dong Z., Yang X., Chai S., Xu Z., Qi D., 2019, in Chinese Control Conference (CCC). IEEE, Guangzhou, China, p. 3023

This paper has been typeset from a TEX/LATEX file prepared by the author.