Retrieval of Images with Coronal Loops from Solar Image Databases

Nurcan Durak
Knowledge Discovery and Web Mining Lab,
Dept. of Computer Engineering and Computer Science, University of Louisville
nurcan.durak@louisville.edu
(Advisor: Olfa Nasraoui)

Abstract

We present a system for the automatic retrieval of images with coronal loops from the solar image database captured by the SOHO/EIT satellite [1]. Our image retrieval system provides relevant data to astrophysicists who need such data to perform studies that shed light on the mysteries behind solar activities and to analyze the relations between coronal loops and other solar events. As part of building this system, we investigated various image preprocessing techniques, image based features, and classifiers to detect coronal loops automatically and to indicate their locations on the images. Despite many challenges related to the coronal loop characteristics, we obtained promising results, namely 85% precision and 83% recall in loop retrieval.

1) Problem and Motivation

The Sun, the source of our life, is a highly energetic star where several gigantic energy revealing events occur. Some events such as coronal mass ejections or the solar wind affect the Earth and might cause damage to polar grids and satellites. Figure 1 illustrates several satellites monitoring the Sun and how a coronal mass ejection affects the Earth. Several satellites have been deployed to closely monitor the solar events, understand their dynamics, and take precautions to reduce their damage on Earth. These satellites, which include SDO, SOHO, TRACE, and YOHKOH, have been taking pictures of the Sun regularly and storing the images in public databases [1]. SOHO is the oldest satellite and has taken more than 30000 images that are stored in the SOHO online database [1].

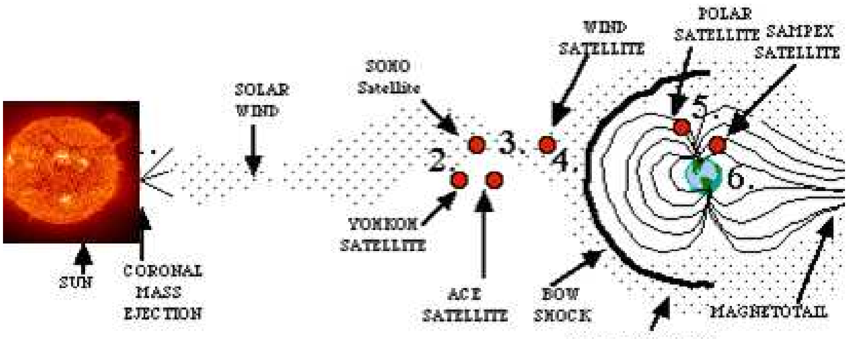


Figure 1 **The interaction between the Sun and Earth along with the designated satellites [10]**

One of the key solar events is coronal loops which are immense arches of hot gas on the surface (corona) of the Sun as shown in Figure 2. Coronal loops are linked to some other events such as the solar flare triggering or coronal heating problem. In order to make progress in studying coronal loops, scientific analysis requires the data observed by the instruments such as SOHO/EIT [1]. Currently, astrophysicists prepare the data set containing coronal loops by looking at every single image separately and putting markings around the coronal loops manually. The manual search for coronal loops is not only subject to human error but is also time-consuming and tedious. The main contribution of our study is developing an image retrieval tool that can (i) separate images with coronal loops from images without coronal loops; (ii) extract a subset of images that contain loops from a larger collection, and locate the detected coronal loops on these images. We are especially interested in coronal loops that are visible outside the solar disk. For this study, we focused on coronal loops that have strong arches and are easily discernable from the background.

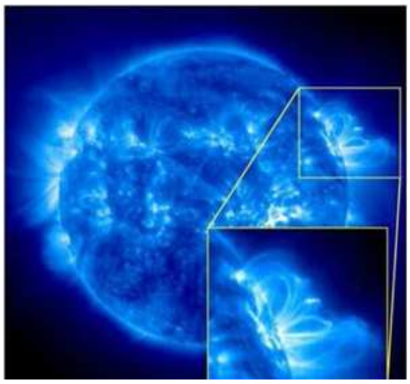
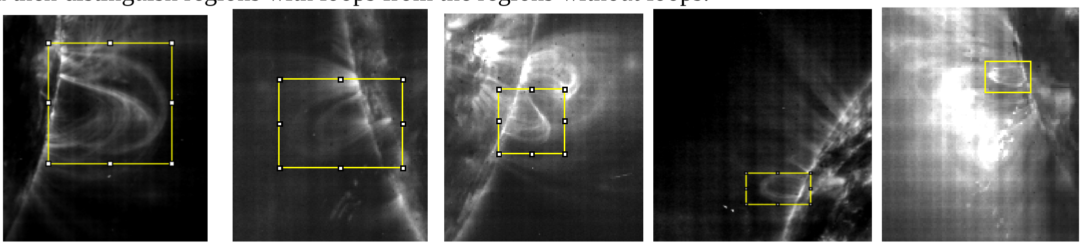


Figure 2 **SOHO/EIT Image with a Coronal Loop Region on the outside of the Sun**

2) Background and Related Work

Coronal loop detection from solar images has sparked attention from different aspects [2, 3, 4]. The main aim of these studies is highlighting the loop structures on the given solar image regions by using curve tracing [2], ridge detection [4] or the wavelet transform [3]. Most of these algorithms, however, were tested and compared on images that were *already known* to contain loops. The main difference between our work and these previous studies is that our methods are not aware of the existence of loops in the given images, while existing methods already know that a given image contains a loop. Thus, our automated methods must first have to learn the characteristics of coronal loops and then distinguish regions with loops from the regions without loops.



(a) **(b)** **(c)** **(d)** **(e)**

Figure 3 **Coronal loops are marked by experts. Coronal loop characteristics vary significantly from one loop to another. Finding common features to represent them is challenging. (a) A big loop (b) A vague loop (c) A noisy loop (d) A small loop (e) A loop interfering with another solar event (prominence)**

In this study, we faced the following challenges:

1. *Finding the most appropriate image preprocessing sequence*: Coronal loops can be embedded into bright regions and the densities along the coronal loops may change. Without hurting the general structure of the coronal loops, we desire to bring out the loops from their surroundings and suppress the other solar events as much as possible.
2. *Extracting principal contours:* We would like to obtain each individual coronal loop as a single contour. However, extracting individual contours is difficult due to the following reasons: coronal loops may intersect each other or with other events as shown in Figure 3(a), 3(c) and 3(e); the loop structures can also be surrounded by noise points; loops do not reflect the same intensity level along the entire arch as shown in Figure 3(a); some loops could be too faint as shown in Figure 3(b), and after preprocessing, some loops could disappear; some loop segments can contain missing parts. Curve tracing algorithms [5, 6] may choose a noise point or a wrong point instead of a loop point, and this leads to undesirable contours.
3. *Finding the most appropriate feature set*: Finding common features for all kinds of coronal loops was another challenge. Coronal loops generally have asymmetrical semi-elliptical or arched shapes. Their sizes, orientations, and arc heights vary from one loop to another, as shown in Figure 3. In addition to coronal loops, several other solar events (e.g. prominence, flares) have similar shapes to loops, e.g. Figure 3(e). These problems make the feature extraction phase more critical.

3) Approach and Uniqueness

To classify solar images according to the presence of coronal loops, FITS image files were downloaded from the EIT online database [1]. These images are 1024x1024 in size and consist of gray level intensities. The training data set was initially prepared by astrophysics experts who marked each coronal loop in the downloaded solar images by enclosing it within a minimum bounding rectangle as shown in Figure 3. The images are then prepared to bring out the coronal loops and to suppress the other events as much as possible. Then we extract a strip around the Sun and extract salient contours from the strip. We label every contour as "*Loop*" or "*Non-Loop*" and then extract geometric features of the labeled contours to train classifier models. Using these classifier models, we implemented an image retrieval tool that can detect coronal loops from an independent test set. The architecture of the general system is illustrated in Figure 4.

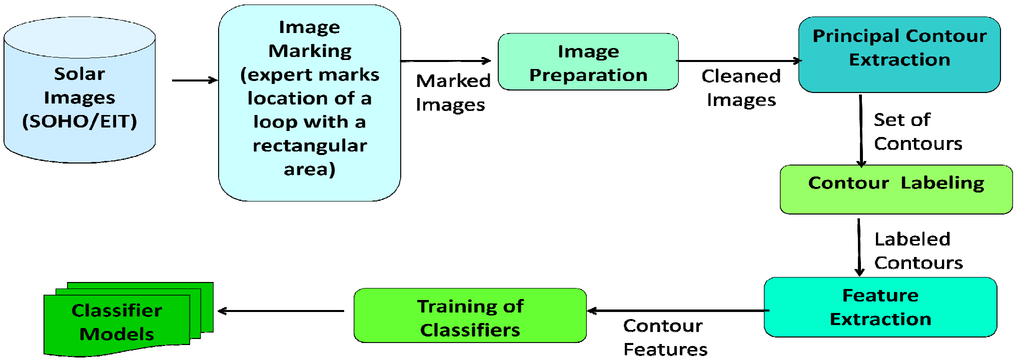
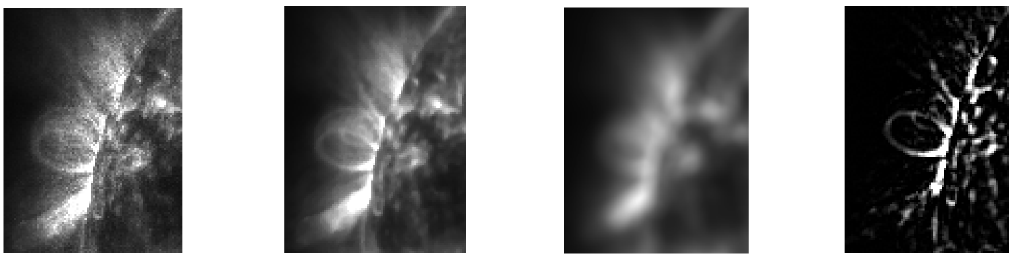


Figure 4 **General Structure of the System**

***3.1 Image Preparation**:* The images are cleaned by using the solar software (ssw) library [7] to get rid of instrumental defects and grid artifacts which are clearly visible in Figure 3 (d) and (e). A portion from one of the cleaned images after ssw is shown in Figure 5 (a). These images contain some specks and pixel level salt and pepper noise. We clean the image by using median filtering to remove specks and by using Gaussian filtering to clean the salt and pepper type noise. Figure 5 (b) shows an example of a filtered image.

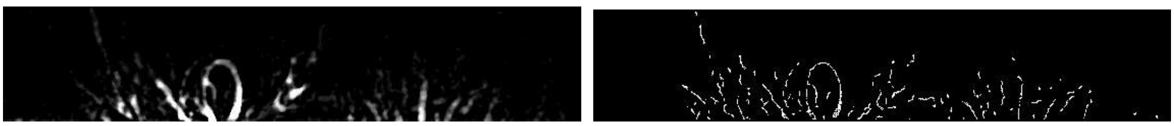


(a) (b) (c) (d)

Figure 5 **(a) Raw Image, (b) After despeckling and Smoothing, (c) Background Image (d)After Background Extraction**

To bring out the loop structures from their surroundings, we perform a background subtraction method, which extracts a background image from the original image. We create the background image by performing the Wavelet transform using the Symlet family with order 4, with soft thresholding, and 40% retaining coefficients. Figure 5 (c) shows the created background image and Figure 5(d) shows the final image after background subtraction.

Since we are interested in the loops outside the solar disk, we extract an image strip (see in Figure 6 (a)) from outside the solar disk by using an angular transformation. Suppose that R_0 is the radius of the solar disk, x_c and y_c are the central coordinates of the solar disk, and H is desired the height of the strip. Then, we create a strip of size $H.2\pi R_0$.



(a) (b)

Figure 6 (a) a portion from the strip generated outside the Sun, (b) the same strip after binarization

From this strip, we prefer to keep the central points of the fluxes instead of all gray values to reduce the system complexity and increase the loop detection speed. We obtain the central points of the fluxes by comparing the intensity value of a point to its four cross-pair neighbors. If the intensity level of a point is equal or greater than at least its two different cross-pairs, then we consider the point to be a central point, otherwise we eliminate the point [8]. Figure 6(b) illustrates the structure image obtained by preprocessing Figure 6(a).

3.2 Extracting salient contours: Even though we obtain much cleaner images after the image preparation stage, we still need to extract salient contours separately and eliminate short independent segments in the strip. As mentioned before, loop segments could be fragmented. The Human eye can easily complete the gaps in related line segments, however this is very challenging for an automated technique. In particular, if other forms intersect with the fragmented loop, then favoring the wrong line segment over the right one is highly possible. Figure 7 (a) shows a sample region obtained from the previous stages. Figure 7 (b) shows the desired loop contour to be extracted from the region, while Figure 7 (c) shows an undesired contour, yet one that is likely to be extracted. The accuracy of the coronal loop detection system depends on extracting the salient contours accurately from the clutter. To overcome these problems, we propose a Salient Contour Extraction method that extracts connected components as an initial hint. Then, we run our specialized curve tracing method which handles gaps and follows the correct path at the junctions [8]. To test how well our Salient Contour Extraction method detects the desired contours, we applied the algorithm on 100 loop contours and 400 non-loop contours which are embedded in cluttered regions. Our technique successfully extracted 88% of the loop contours and 90% of the non-loop contours. The algorithm extracts each salient contour separately, then experts label them as *"Loop"* or *"Non-Loop."* These labeled contours are subsequently analyzed to extract useful features in the training stage.

(a) (b) (c)

Figure 7 (a) Input region, (b) desired salient contour (c) possible undesired contour

***3.3 Finding the most appropriate feature set*:** To decide whether the given contour is a loop or not, we extract geometric features of the labeled contours, then we build a learning model. To calculate the arch height, curvature and linearity of the contour, we use the point to chord distance in Equation (1). Let n be the number of points in the contour, $\mathscr{P}$ the point set of the contour, L the chord between the end points of the contour, d the set of distance values. Given $\mathscr{P}_i = (x_i, y_i)$, $\mathscr{P}_1 = (x_1, y_1)$ and $\mathscr{P}_n = (x_n, y_n)$.

$$d(\mathscr{P}_i, L) = \frac{(y_1 - y_n)x_i + (x_1 - x_n)y_i + (x_1 y_n - x_n y_1)}{\sqrt{(x_n - x_1)^2 + (y_n - y_1)^2}} \quad (1)$$

We extract the following features from the contours.

- <u>*Linearity*</u>: The ratio of the close points to the chord (a distance less than τ) to n gives the *Linearity* value of the contour.
- <u>*Pseudo-curvature*</u>: We calculate the pseudo-curvature by dividing the arch height of the contour by the chord length. The chord length is the Euclidean distance between the endpoints of the contour, and the *arch height* is the maximum distance between the contour and the chord.
- <u>*Elliptical Features*</u>*:* We perform direct Least Squares fitting [9] on the contour to obtain the parameters of the best fitting general conic equation which is $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$. Based on the estimated parameters, we compute the major axis length K and the minor axis length L [9]. Then, we calculate the *eccentricity* of the contour, *the ratio of minor axis to major axis (K/L)*, and the *error of fit ratio* which is the ratio of the number of points with small fitting error to n. The Error of fit was calculated for each point using the gradient normalized Algebraic distance.
- <u>*Smoothness*</u>: We calculate the smoothness of the contour by dividing the contour into small windows and calculating angle changes among consecutive windows. Rapid angle changes between consecutive windows are an indication of *corner points*. The *smoothness* is computed as the root mean square of the angle changes among the windows.
- <u>*Proximity:*</u> Since we allow gaps in the contour extraction method, the contour points might be further away from each other. However, if there are many gaps along the contour, this decreases the proximity of the contour as a

loop. Therefore, we calculate the *proximity* of a contour by averaging the Euclidean distances among the consecutive points.

Our final feature set consists of *linearity, pseudo-curvature, arch-height, eccentricity, minor axis length, ratio of minor axis to major axis, error of fit ratio, smoothness, corner points, and proximity.*

***3.4 Training Classifier Models to Find the most appropriate feature set*:** In the training stage, we extracted the features defined above from 150 loop contours and 250 non-loop contours. Then, we trained an Adaboost learning model based on the C4.5 decision tree learner that scored 85% precision and 83% recall on average in ten-fold cross-validation experiments (We chose Adabooost based on the results of extensive experiments using many different classification methods). Figure 8 illustrates the ROC curve of the model. Our results validate the success of our choice of features and processing method, as well as validate the proposed contour extraction method to solve the coronal loop detection problem.

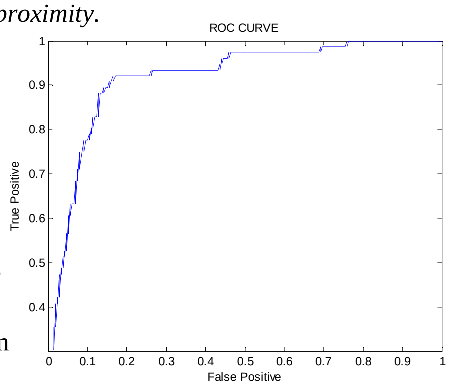


Figure 8 **ROC curve of the Adaboost classification model**

4) Results and Contributions

We developed an image retrieval tool (as in Figure 9) to separate images with coronal loops from images without loops using JAVA. The user can upload a batch of solar images, and the system retrieves a subset consisting of only the images that contain coronal loops. The tool also shows the location of the detected loops on the images. The users are able to browse the result set and save the images.

For each image in the uploaded data, we follow the steps described in Section 3.1, Section 3.2, and Section 3.3. We apply the Adaboost model on the features computed from the contours extracted in the strip located outside the solar disk within an image. If any extracted contour is classified as "*Loop*" by the model, we assume that the image has at least one loop and we locate the "*Loop*" contours on the image.

We tested our model on an independent test set that consists of 50 images with coronal loops and 50 images without loops. The system separates the images with 90% accuracy.

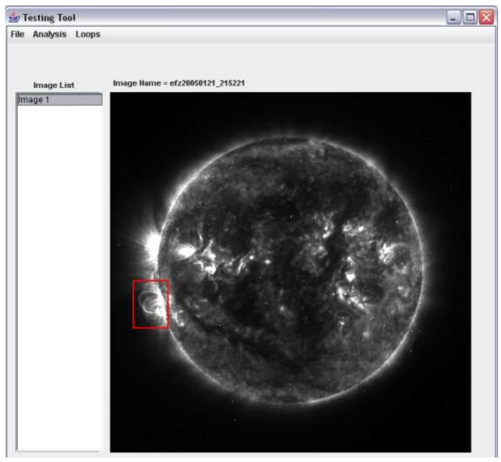


Figure 9 **Image Retrieval Tool**

Contributions: 1) We proposed a Salient Contour Extraction method which deals with the noise and broken contours in cluttered images. Our evaluation on an independent test set with real contours embedded in noisy regions yielded a correct detection rate of 88% of the desired contours, while ignoring the undesired contours. Our contour extraction method can be applied to any cluttered images to extract elliptical arcs. **2)** To bring out the coronal loops, we investigated many different image processing techniques and presented the best image cleaning sequence that conserves the loop structures while suppressing the other solar events. **3)** We experimented with several image based features and selected the feature set giving the highest precision and recall in the training phase. We classified the contours as *"Loop"* or *"Non-Loop"* based on their geometric features with 85% precision. **4)** We implemented an image retrieval tool to separate images with loops from the rest, and this tool was tested and validated by astrophysicists.

Acknowledgement

This work is supported by the National Aeronautics and Space Administration under Grant No. AISR-03-0077-0139 issued through the Office of Space Sciences and by National Science Foundation Grant IIS-0431128.

References

[1] EIT: Extreme ultraviolet Imaging Telescope on board Solar and Heliospheric Observatory (SOHO): http://umbra.nascom.nasa.gov/eit/

[2] J.K Lee, T.S. Newman, G.A. Gary. *Oriented connectivity-based method for segmenting solar loops.* Pattern Recognition, 2006.

[3] M.J. Aschwanden, J.K. Lee, G.A. Allen, M. Smith, B. Inhester, *Comparison of Five Numerical Codes for Automated Tracing of Coronal Loops*. Solar Physics, Volume 248, Issue 2, 359-377, 2007

[4] B.Inhester, L. Feng, T.Wiegelmann, *Segmentation of Loops from Coronal EUV Images*, Solar Physics, 248, 379–393, 2007.

[5] K. Raghupathy, T. W. Parks, *Improved Curve Tracing in Images*, ICASSP (2004), 581-584.

[6] C. Steger, *An unbiased detector of curvilinear structures*, IEEE Trans. On Pattern Analysis and Machine Intelligence, February, (1998), 113-125.

[7] SolarSoft: www.lmsal.com/solarsoft/sswdoc/sswdoc

[8] Nurcan Durak, Olfa Nasraoui: *Principal Contour Extraction and Contour Classification to Detect Coronal Loops from the Solar Images*, ICPR 2010

[9] A. Fitzgibbon, M. Pilu, and R.B. Fisher. *Direct Least Squares Fitting of Ellipses*, IEEE Trans. of PAMI vol.21 no.5, May 1999.

[10] The Sun-Earth Connection: http://edmall.gsfc.nasa.gov/99invest.Site/SUN-EARTH1/Sun1.html