Automatic Analysis of SOHO Images

Emanuele Zattin

Kongens Lyngby 2007
IMM-PHD-2007-97

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Kongens Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

IMM-PHD: ISSN 0909-3192

Abstract

With the project the author extends a previous work about detecting and tracking sungrazer comets in a sequence of astronomical images taken by the instruments mounted on the SOHO satellite. Purposes, design decisions, methods applied and formal explanation will be provided. Good results are obtained and will be analyzed and criticised in order to allow further future improvements of the algorithm. The results will demonstrate that comet hunting in such images, which is now only redacted by amateur astronomers who visually analyze the pictures, is doable in a human unaided way given a previous proper choice of the parameters. The implementation will be developed using the Python programming language with the aid of mathematical libraries such as Numpy and Scipy.

Preface

This project is the development and maturation of an idea that struck me more than five years ago now. I first came across SOHO hunters reading an article on an astronomy web-zine and immediately I started to download images looking for my own comet. After a while I thought I found something interesting and sent a report. It was May the 16th of 2002. That report turned out to be totally inappropriate since the "object" i pointed out was changing in intensity, shape and speed in order to be considered a comet. The community of astronomers were anyway kind enough to tell me where i was wrong and pointed me towards the right direction. My first thought was:

"Wow, this is going to be hard!"

I think I have always been kind of a lazy person. Not the work-shy kind of lazy, but the kind that will make me try to find an alternative way to do a repetitive and boring task. So my second thought:

"Can't a computer do this for me?"

At that time the answer was negative. I did not have the necessary knowledge and preparation and slowly that idea sank in my mind.

Two years ago i came to Denmark and started to study in DTU and right after my first course about Computer Vision and Image Analysis that idea reemerged. From that moment on I started to work on it and the what follows is the result.

Acknowledgements

Many people helped me during the preparation to and development of this project.

Of course the first on the list is my supervisor, Henrik Aanæs, who immediately showed interest in my idea and paved my way to move from ideas to facts. His great preparation and broad knowledge was a great source of informations that helped me to make my mind clear and several cases. I also thank him for giving me the opportunity to show my ideas and results also to astronomers of the Niels Bohr Institute.

Credits also go to Jens Michael Carstensen who accepted to include my project as a candidate for a three week course. He shown interest and provided me useful acknoledgements and tips.

A special thank goes to Hans Brun Nielsen who was a real source of inspiration. His massive experience and preparation can only be compared to his humanity and humility.

Lorenzo Bolla also needs to be on this list. He is probably the smartest guy i know and his tips were simply unvaluable.

Contents

Abstract **i**

Preface **iii**

Acknowledgements **v**

1 Introduction **1**

2 Historical Background **3**

3 Previous Results by the Author **5**

3.1 Preamble 5

3.2 The Algorithm 5

3.3 Implementation 11

3.4 Results 11

3.5 Test field 14

4 Rewriting the application in Python **15**

4.1 Motivations 15

4.2 The Scipy Project 16

4.3 How Python solves the problems 18

5 Image cleaning **19**

5.1 Image blurring 19

6 Finding Objects **23**

6.1 Finding Local Maxima 23

6.2 Distribution of the Regional Maxima 25

6.3 Removing False Maxima 28

7 Computing the Properties of the Objects **31**

7.1 Sup-pixel Precision Center of the Object 32

7.2 Estimated Maximum 32

7.3 Central Intensity 33

7.4 Sharpness 33

7.5 Elongation 34

8 Tracing the paths **35**

8.1 Object Tracing vs Path Tracing 35

8.2 First level constraints 36

8.3 Second level constraints 37

8.4 Filtering out stars 37

9 Results **39**

9.1 Implementation Notes 39

9.2 Analysis of the 2006 data-set 40

9.3 Possible future developments 42

10 Conclusion **45**

A Source Code **47**

B Log Files **65**

B.1 S1 65

B.2 S2 81

Credits **91**

Chapter 1

Introduction

This project was thought and developed in order to answer the question: can a computer, a mathematical model, a set of algorithms substitute the patient and time-consuming work of several people in analyzing a set of astronomical images in order to detect and track comets?

This question might appear very specific, but it involves bigger and more important considerations. The introduction to scanners first and then CCD sensors brought a great revolution in many fields. One of these is astronomy.

Nowdays the output of every telescope and astronomical intrument (depending on the wavelength range of electromagnetic waves it analyses) is an image or a set of images. The amount of data available is so big that at the present time astronomers all over the world are struggling to keep the pace and avoid to loose important data.

The not even too far future seems to bring even bigger challenges since the amount of data to analyze is growing exponentially as the size of sensors grows and lenses are able to detect fainter and fainter objects. We are soon reaching the point where either we will be able to develop models and programs to perform the big part of the analysis or we are going to loose precious informations for ever since even the storage of this astonishing amount of data will become at some point impossible.

These consideration puts the above stated question under a different light. Answering that is a tiny contribution to a huge challenge, but still, as 2500 years ago Confucio said "Even mountains are made of grains of sand".

Chapter 2

Historical Background

SOHO is a cooperative mission between the European Space Agency (ESA) and the National Aeronautics and Space Administration (NASA). SOHO studies the sun from deep down in its core out to 32 solar radii. The spacecraft orbits around the Earth and from this orbit, SOHO is able to observe the sun 24 hours a day. It carries twelve state-of-the-art instruments expected to meet the mission's three principal scientific objectives. These objectives are: to study the solar interior, to study the heating mechanisms of the solar corona, and to investigate the solar wind and its acceleration processes.

Even though SOHO's primary objectives relate to solar and heliospheric physics, the onboard LASCO instrument has become the most prolific comet discoverer in history! LASCO is a three-coronagraph package (C1, C2, & C3) with nested field of views of 1.1-3, 2.5-6, and 4-32 solar radii, respectively. The C1 instrument was designed to observe hot coronal emission and, therefore, has not observed any comets. However, since LASCO began taking observations in January of 1996, the C2 and C3 coronagraphs have observed over 1200 new comets and several known comets. In addition, the SWAN instrument onboard SOHO has discovered 4 new comets and observed many known objects. The UVCS instrument has also observed a handful of known comets.

The kind of comets visible by these instruments are called sungrazers. Though there is no official definition of a sungrazer, it refers to a comet that passes very

close to the Sun. It is most often used to describe comets of the Kreutz group, which have included some exceptionally bright daylight comets as well as the stream of tiny cometary fragments found in SOHO. As a descriptive term, a few other comets of very small perihelion distance such as the Great Comet of 1680, which passed a mere 200,000 km from the Sun's photosphere, deserve the title of sungrazer. Non-Kreutz comets found in SOHO are sometimes referred to as sungrazers, though such comets with less extreme orbits are often called near-Sun comets (and occasionally, sunbathing comets).

About 85% of SOHO comets belong to the Kreutz group, a family of comets whose members all travel in similar orbits. It is named for Heinrich Karl Friedrich Kreutz, (1854-1907), an astronomer at the University of Kiel who investigated comets that passed close to the Sun. He noticed similarities in the orbits of comets that appeared in 1843, 1880, 1882, and 1887, and determined that they were probably all once part of a larger body; he suspected that comets seen in 1668 and 1702 also belonged to this group.

Since 2001, the vast majority of SOHO comets have been found by amateurs; the rest were found by SOHO staffers. LASCO images are available for download by the public (at the same time they become available to scientists), primarily at http://soho.nascom.nasa.gov/data/realtime-images.html.

Despite the large number of SOHO comets that are discovered (an average of more than 100 a year since the spacecraft's launch in 1995) and the impressive totals racked up by some experienced hunters, it is not easy to be the first to report one, particularly for newcomers. (Even the most skilled hunters sometimes go for months without finding one.) There is a learning curve to detecting the often subtle signs of a developing SOHO comet (which often doesn't look cometary by standard measure), and to becoming familiar enough with their appearance and motion to be able to report them before other hunters. (There is a core group of about a dozen skilled and dedicated hunters who report these comets within hours, often within minutes, of when they first become confirmable.) You need to learn to quickly identify a comet and distinguish it from cosmic rays, noise, and other artifacts that may mimic a comet, measure its position (in x,y coordinates) on a Series (preferably a minimum of 4) of images, and then file a report using the report form. Mastering these skills can take many months. Many of the comets, particularly ones in C2, are found in the unprocessed black-and-white images, the first to be posted, which appear on the LASCO/NRL site, and are not archived. Working with them requires that you download the images quickly, or you will miss them.

Chapter 3

Previous Results by the Author

3.1 Preamble

This project takes as a starting point the results achieved during a special course held in DTU during the autumn semester in 2006. In this chapter the methods and the results obtained will be introduced and discussed in order to give a general overview of the tasks met and the relative solutions which were applied. It has to be noticed that that project has been developed without any bibliographical research and that the solution found are simply the result of applying known techniques.

3.2 The Algorithm

The cameras providing the images were designed to study the corona of the sun. This means that the emphasis is of course put in showing the details of the evolution of the corona and not in the stars in the background. What happens in SOHO comet hunting is totally the opposite: the corona is some noise that needs to be get ridden off while the stars (or what appear as such) contain the

needed information. These could be in fact stars, asteroids, comets or, in most cases, simply cosmic noise or artificial satellites. Once the objects present in the images are detected a tracking system will be developed that will consider all the possible combinations of paths for each object and filter out the non plausible ones. The whole process can be summarized in three steps:

- Cleaning the images
- Detecting the objects
- Computing the plausible paths

For each item some details will be now given.

3.2.1 Cleaning the images

Without a good cleaning of the original images the algorithm would have to deal with much more information making the path detection part much longer. The main task in this process is to get rid of the corona from the images since it contains no useful information for the purpose of the project. Visually analyzing the pictures it is easy to notice that the corona is something that changes smoothly along the image so a plausible approach would be to remove low frequency information from the signal. Unfortunately this is not possible in this application since comets do not always appear at high frequency signals in the images (point light source) and sometimes they appear like a small blurry object (Figure 3.1). This means that if we got rid of low frequencies we would end up erasing possible comets as well.

Looking at single images did not bring to any better idea, but drifting along some consecutive images it was easy to notice that the corona hardly changed over small time lapses. This fact suggested the idea to do a subtraction between two consecutive images:

$$I_p^F = \max(I_p^A - I_p^B, 0), \forall p$$

where p is a pixel of the image defined by its coordinates. The max operator makes sure that the resulting image I^F will not contain information about the objects in I^B. This subtractions brings to several advantages:

- removal of most information about the corona

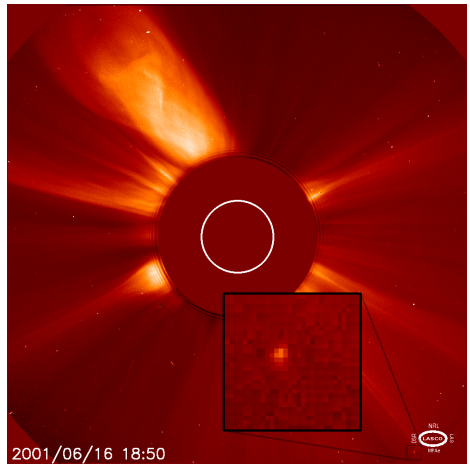

Figure 3.1: Example of blurry comet

- removal of constant noise such as dust on the objective and imperfections of the CCD sensor of the camera
- the resulting image will have a nice black background which increases the contrast.

The drawback of this method is that the longest the interval between two images is, the more the difference between the corona in the two images will affect the result.

An example of this process is shown in Figure 3.2.

At this point the image is good enough to try to extract the position and the intensities of the objects. In this case it is convenient to operate in a multi-scale context since images still contain some very high frequency noise i don't want to deal with so the first task is to apply a slight Gaussian blur to the whole image. The value of σ must be high enough to get rid of the noise, but small enough not to loose any information about the objects. The kernel of the filter must be a square formed by edges of odd number of pixels.

Another useful thing to do is to stretch the color spectrum of the images in order to increase the contrast. It is important to notice anyway that this operation must be done with the same parameters for all the images of the sequence otherwise the estimation of the intensity of the objects would be uneven and this must not happen since it will be one of the criteria do filter good objects

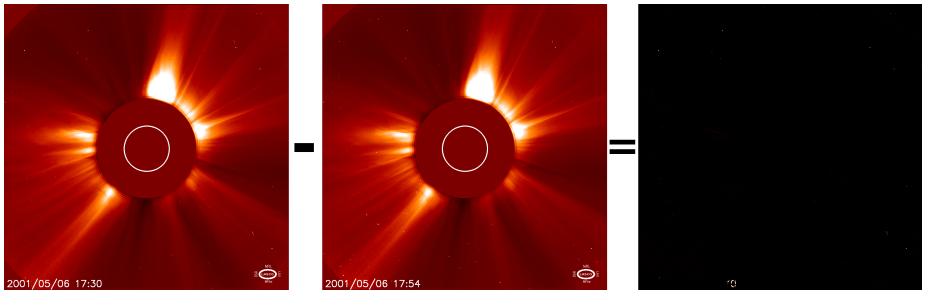

Figure 3.2: Difference between two consecutive pictures

from bad ones.

3.2.2 Detecting the objects

Now the images are clean enough to detect object. The first approach tried is blobs detection as described in [4] and this turned out to work fairly good for circular objects, but some of the detected comets also present a tail that makes the algorithm fail.

After many tries the most solid technique resulted to be regional maxima detection: both point-shaped object and tailed comets get spotted correctly. Even for comets with a tail the algorithm detects the head correctly which allows to trust the color intensity value. This method is based on the morphology theory and was first proposed in [8]. It consists in iterating a dilation operation on the original image subtracted by 1 using as a boundary condition the original image. More details will be unveiled later.

3.2.2.1 Data structure

For each image the detected objects are stored in lists and each element contains an index which identifies the object in that particular image, the coordinates in the image space and the intensity in that position. An example is shown here below.

index	x	y	value
1	125	206	97
2	32	342	23
3	234	421	134
⋮	⋮	⋮	⋮

3.2.3 Computing the plausible paths

At this point all the positions and intensities of all the objects are retrieved for every image. This data will be used to detect any possible comet present in the sequence and, considering that in every picture an average of 150 objects is found, it means that the algorithm will have to consider about 500 millions possible combinations. This involves a huge amount of computation time so i need to consider a technique to lower the number of combinations in a smart way.

The way i decided to go is to divide the process into steps, each of which will consider a transition from one image to the following one. The trick is that, when processing the following step, to consider only the combinations that proved to be plausible until that point.

3.2.3.1 The first step

The first step takes into consideration what changes in the first two images of the sequence. As a first thing i was to filter out stars from the object lists of both images since they are easy to spot. They will always move from left to right with a more or less constant speed. Actually when the sun is far from the sky equator it is possible to notice some differences in the amount of pixels the stars move in the top and the bottom of the image so it is wise to design the star detection algorithm in a pretty “loose” way.

After this is done i go throw all the remaining objects in I^A and check all the possible combinations with the objects in I^B. I just want to consider objects that move of a certain amount of pixels (depending on field of view of the telescope i am taking the images from) and that have a pretty similar pixel intensities. One more rule the object should follow to be a candidate comet is that during the sequence it should move towards the sun.

To save the relevant information i decided to use the following data structure:

i_A	i_B	d_x	d_y	d_v	v	d

where:

- i_A: index of the object in I_A
- i_B: index of the object in I_B
- d_x: difference of x coordinate of the object between the two images
- d_y: difference of y coordinate of the object between the two images
- d_v: difference of intensity value of the object between the two images
- v: speed of the object (the length of the speed vector)
- d: direction of the object (the angle component of the polar coordinates of the speed vector)

Expressed in an analytical way this phase is described as:

$$\begin{aligned} min_x < d_x < max_x \\ min_y < d_y < max_y \\ d_v < max_v \end{aligned}$$

where $min_x, max_x, min_y, max_y, max_v$ are parameters known "a priori".

3.2.3.2 The following steps

The steps after the first one will operate in the exact same way but will also add some more tests in order to make the classification more precise.

First of all comets move with constant speed so the algorithm will take care to filter out the objects moving irregularly. The second decision criteria is the direction: a comet will move toward the sun with a slight curvature so the process needs to detect and filter out "S-shaped" movements.

In an analytical way:

$$\begin{aligned} \|v_{AB} - v_{BC}\| < max_v \\ \|d_{AB} - d_{BC}\| < max_d \end{aligned}$$

3.3 Implementation

The model was implemented in Matlab and the source code is available in the Appendix. The biggest challenge during the development was the lack of advanced data structures in the framework. In the end a workaround was found but the code readability pays the price. Matlab is a great tool to test IF an algorithm works and, concerning this, it really did a great job. Of course speed is not one of the features of my implementations, but once the algorithm is working and tested then another implementation will be possible using another programming language.

An interesting note should be done apart from one of the bottlenecks of the application: the test of plausible paths. It is, as a matter of fact, a research in a 3-dimensional space (d_x, d_y, d_v) and right now it is computed in a very straightforward way. This could be made much faster using some smart data-structures such as k-d trees which allow a precious performance increase for big data-sets.

3.4 Results

The results of the model are to be considered satisfactory. During the development process a small set of test scenarios has been used, both containing and not known comets. This was an invaluable help during the parameters estimation process and gave very good results with further tests. The first set contains a known comet found in 2001 and of which several images are available. Already using the first four the algorithm was able to exclusively spot it. The second test was performed on a blurry comet of which only three images are available. Even this time the algorithm succeeded and gave the expected result. The other test images contained no comets and were especially used to adjust the parameters and get as few false positives as possible. The program takes as input four consecutive images and gives as output the number of candidate comets found and, for each of them, a picture showing the path performed by the object.

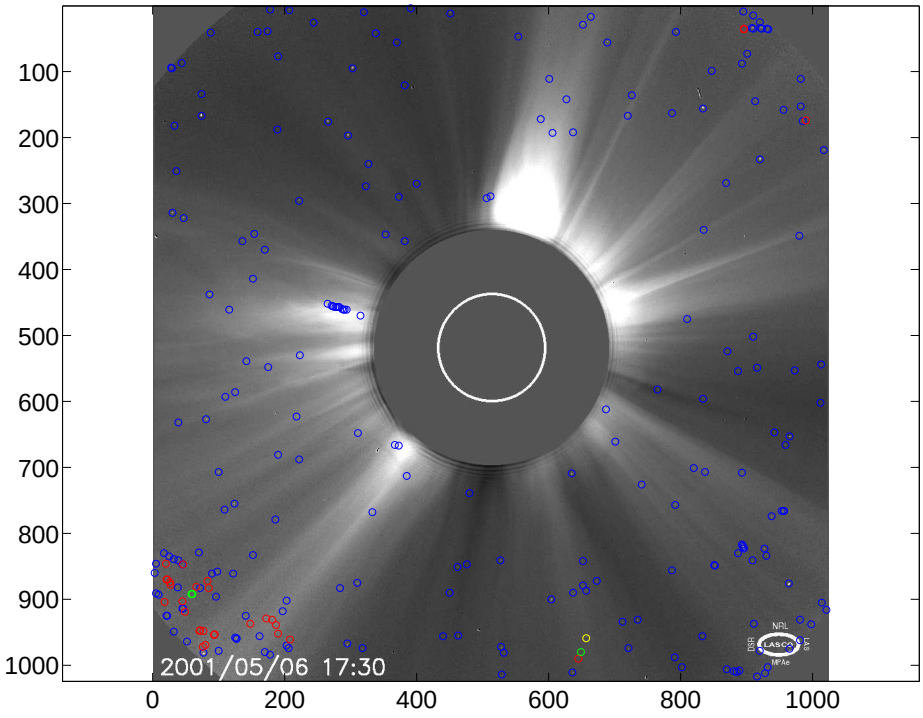

Figure 3.3: Detection of a comet visible in the bottom of the image. Four images have been used and the time-lapse between the images is not constant. This is the first comet ever captured by this algorithm.

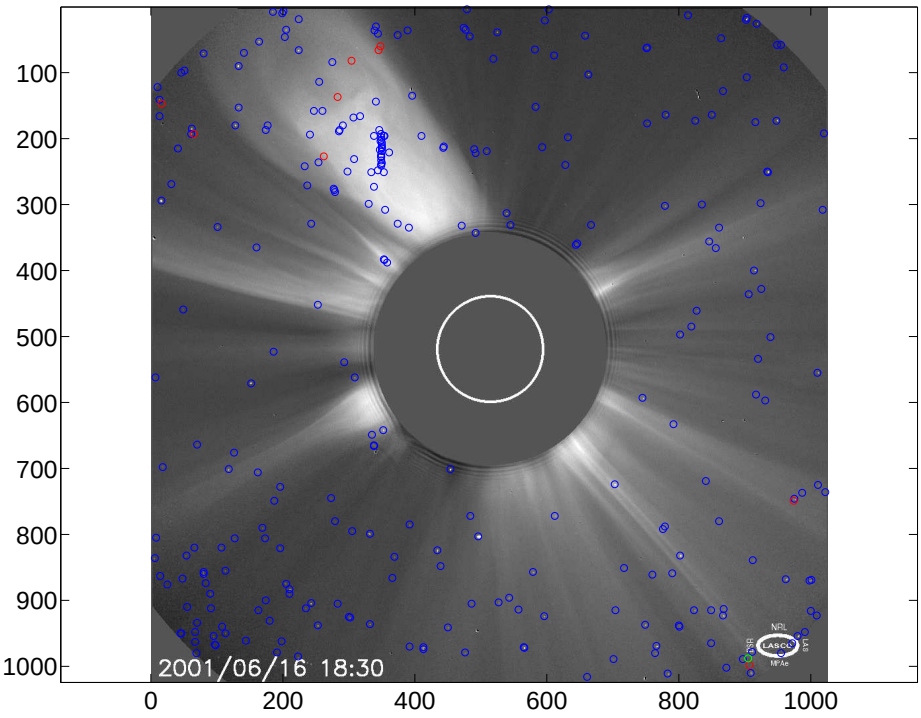

Figure 3.4: Detection of a comet visible in the bottom of the image. Two factors make this picture interesting. First of all only three images were necessary to spot the comet and secondly the comet is not point light-source but a blurry object several pixels wide.

3.5 Test field

A script has been developed to download all the images from the 2006 archive and scan it for comets. The process took about one week and both the script and the results are available in the Appendix.

The program showed a good robustness in terms of accuracy but its performance makes it unusable when a great number of object is found. This can happen mainly for two reasons:

- An explosion on the sun causing a huge and fast corona beam. This will make the subtraction process fails and many object will be found. This can be solved teaching the algorithm to only look for objects in the proximity of the borders.
- A solar particle storm. This natural phenomenon casts a huge amount of CMEs (Corona Mass Ejections) flares and energetic particles resulting in very "snowy" images. When this happens a huge number of objects is found and produces a big number of false positives. This can be avoided putting a limit to the number of objects allowed in a picture. If it gets exceeded then the sequence will not be analyzed.

Analyzing the result a big amount of false positives has been detected which can most likely be avoided setting the parameters more accurately. Also a small number of false negatives has been noticed but this is related to very "uncommon" comets that moved in an orbit that passed fairly close to the earth giving them an apparent irregular speed.

More precisely, in year 2006 45 comets were spotted in the images sent by the C2 camera of the LASCO instrument. This is a narrow field angle camera (compared to the C3) and allows to spot faint objects. Of these 45 objects, 43 were identified correctly and 2 were missed. Investigating the reasons of these false negatives i have found out that they happened when a comet passes very close to a star. The blurring process makes the algorithm believe that only one object is present when there are in fact two. Then the star/comet object is removed from the database when the algorithm filters out the stars. A solution to this problem could be to avoid to delete stars from the database but simply to mark them.

Another result of the simulation is that several dozens of comet reports have been filed by the algorithm. Some of them are clearly false positives, due mainly to asteroids and random noise. Of the remaining objects i chose a selection of 19 interesting ones and sent reports by email to the organization certifying comets.

Chapter 4

Rewriting the application in Python

4.1 Motivations

The previous implementation of the Model was developed in Matlab. Considering the programming workflow

1. Make it work
2. Make it good
3. Make it fast

Matlab for sure excels in the first one. It provides a simple way to implement an algorithm and to test new ideas. On the other hand there are also some drawbacks, the most important of which are:

- price
- code execution speed

- portability
- expandability

Some products, such as Octave or Scilab, offer a valid solution for the price problem, but the code runs even slower than on Matlab. A valid solution is Python, an open source scripting language which is currently being used for a great variety of applications from companies and organizations such as NASA, Google and many more.

4.2 The Scipy Project

Scipy is open-source software for mathematics, science, and engineering. The SciPy library depends on NumPy, which provides convenient and fast N-dimensional array manipulation. The SciPy library is built to work with NumPy arrays, and provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimization. Together, they run on all popular operating systems, are quick to install, and are free of charge.

4.2.1 Numpy

The fundamental library needed for scientific computing with Python is called NumPy. This Open Source library contains:

- a powerful N-dimensional array object
- advanced array slicing methods (to select array elements)
- convenient array reshaping methods

and it even contains 3 libraries with numerical routines:

- basic linear algebra functions
- basic Fourier transforms
- sophisticated random number capabilities

NumPy can be extended with C-code for functions where performance is highly time critical. In addition, tools are provided for integrating existing Fortran code.

4.2.2 Scipy

SciPy is an Open Source library of scientific tools for Python. It depends on the NumPy library, and it gathers a variety of high level science and engineering modules together as a single package. SciPy provides modules for:

- statistics
- optimization
- numerical integration
- linear algebra
- Fourier transforms
- signal processing
- image processing
- genetic algorithms
- ODE solvers
- special functions

and more.

SciPy is developed concurrently on both Linux and Windows and it compiles and runs successfully on Mac, Solaris, FreeBSD, and most other platforms where Python is available.

4.2.3 Matplotlib

Matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. It can be used in python scripts, the python and ipython shell (ala matlab or mathematica), web application servers, and six graphical user interface toolkits.

4.3 How Python solves the problems

4.3.1 Price

Python, Numpy, Scipy, Matplotlib and all the other libraries used are free and released as open source

4.3.2 Code Speed Execution

The Weave component of Scipy makes it extremely simple to include C/C++ code directly into Python functions, without the need of creating a whole module as it happens with Matlab and .mex files.

4.3.3 Portability

Python and all the used libraries are available for all the popular operating systems.

4.3.4 Expandability

Python, being a general purpose scripting language, is provided of a huge amount of libraries for the most different uses. In this particular project it was important to download images from the web and this could be done easily while it was very complicated in the Matlab version.

Chapter 5

Image cleaning

The model consists in analyzing four consecutive images. The first step consists in cleaning the images which, as in the previous implementation, is done subtracting two consecutive images pixel-wise:

$$p_c = \max(p_i - p_{i-1}, 0), \forall p \in I$$

This will provide the clean image I_c. An example of this process is shown in Figure 5.1.

This step does not rely on any parameter so no sensitivity analysis is necessary.

5.1 Image blurring

I_c contains a big amount of high frequency noise that will affect the object detection. The Fourier transform of the previously shown I_c is shown in Figure 5.2.

To increase the efficiency of the object detection process I_c will be convolved with a Gaussian distribution giving I_b as a result. In order to avoid to lose

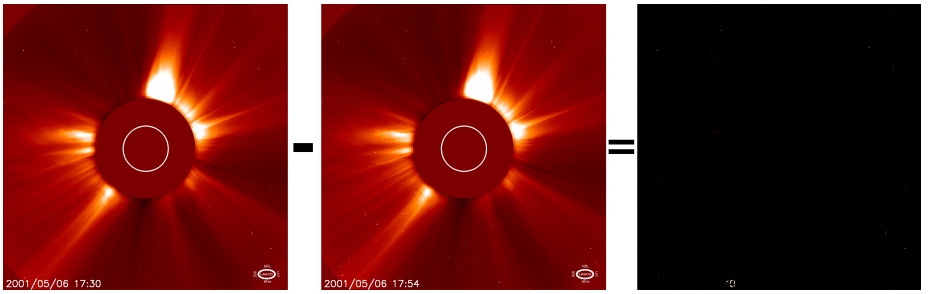

Figure 5.1: Image subtraction process

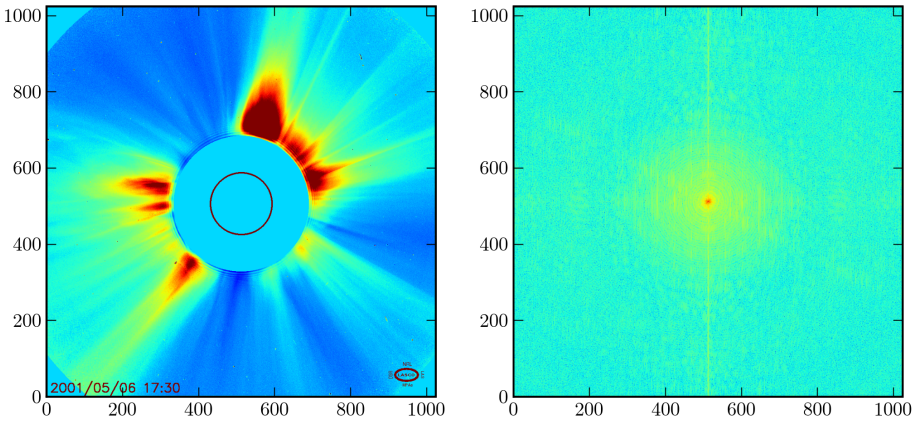

Figure 5.2: On the left a sample image and on the right the magnitude of its Fourier Transform in logarithmic scale (optical Fourier transform)

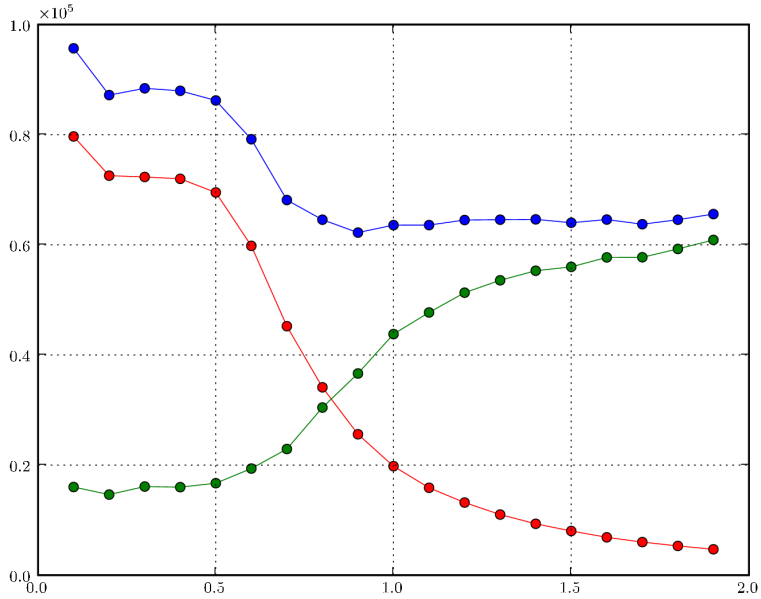

Figure 5.3: How the number of maxima change according to σ. Blue dots represents the number of pixels belonging to regional maxima. Red dots represent the number of objects. The green dots show the difference between the previous two parameters.

important information we need the standard deviation σ of the Gaussian to be less than 1. This will guarantee that objects distant one pixel away from each other will still be distinguishable. The choice of the right σ may change from image to image but it is important to keep it constant during the analysis of the whole four images sequence in order to judge all the pictures with the same criteria and get reliable results. For this model a value of 0.7 has been chosen.

5.1.1 Sensitivity Analysis on σ

Figure 5.3 shows how the number of objects found in the image is dependant from the value of σ.

We can be seen from the plot, as σ gets bigger than 0.7 the number of pixels belonging to regional maxima stays more or less constant while the number of objects decreases dramatically. This means that single regional maxima tend to get bigger and bigger and incorporate each other. Since we want to keep a good granularity and sensitivity over the object, 0.7 proved to be a reasonable choice for σ.

Chapter 6

Finding Objects

6.1 Finding Local Maxima

I_b is now clean enough and ready to be used to detect objects in it. To achieve this a regional maxima search is performed. The regional maxima of a greyscale image is defined as a connected set of pixels of constant intensity from which it is impossible to reach a point with higher intensity without first descending; that is, a connected component of pixels with the same intensity value, t, surrounded by pixels that all have a value less than t. This is different from local maxima since these can only be one pixel big while regional maxima can be extended as shown in Figure 6.1.

Many methods are available to detect regional maxima, the most popular of which is probably the one described in [8] which uses mathematical morphology.

For a greyscale image I_o it is defined as:

$$I_{RM} = I_o - G_r(I_o - 1)$$

where G_r is the morphological greyscale reconstruction by dilation function.

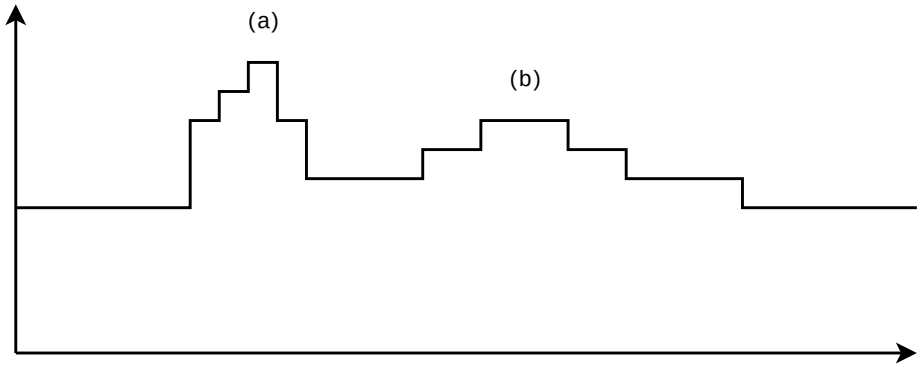

Figure 6.1: (a) is both a local maximum and a regional maximum, while (b) is just a regional maximum because it extends on more than one sampling point

An example of the process is shown in Figure 6.2.

6.1.1 Morphological Reconstruction by Dilation

Reconstruction by dilation is part of a set of morphological image transformations referred to as geodesic. It operates on a "marker" image and a "mask" image and the marker image must be less than or equal to the mask image on a pixel by pixel basis. It is defined as the dilation of the marker image with respect to the mask image iterated until stability.

The greyscale version of the operation was first introduced, formally defined and implemented in [8]. The paper also suggests some implementations, the fastest of which is used by Matlab in the `imregionalmax` function available in the Image Processing Toolbox.

Other algorithms are available, as, for example, the one proposed in [2]. This version is very efficient and it was used in the implementation of this model. The C code is available from the authors of the paper and released with a licence that allows to reuse the code. This made the inclusion in the Python code very simple thanks to the Weave library.

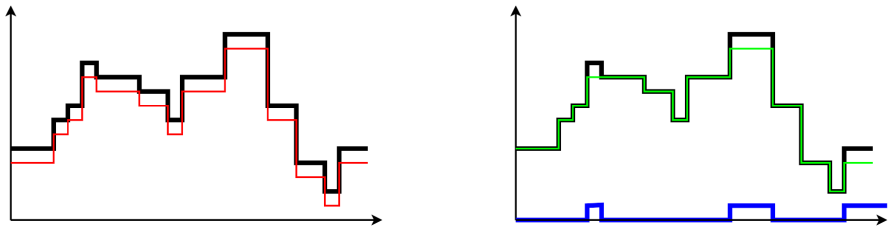

Figure 6.2: Regional Maxima in the 1D case: on the left the black line indicates the original signal I while the red line indicates $I-1$. On the right the red line indicates the reconstructed signal $G_r(I-1)$ while the blue line indicates the binary regional maxima signal obtained from $I - G_r(I-1)$

6.2 Distribution of the Regional Maxima

The algorithm now performs a search for regional maxima and the output is a binary matrix that contains ones in correspondence of maxima and zeros elsewhere.

Since the image still contains some noise it is interesting to see how the maxima are distributed according to their intensities. The computation power and memory amount are limited so the model will need to consider this boundary and concentrate on maxima that lie above noise-level.

The histogram in Figure 6.3 shows how regional maxima are distributed according to their pixel intensities in a typical image. Unfortunately the position and size of the bump changes from image to image so it is not possible to simply set an intensity boundary under which not to consider maxima valid for all pictures.

To fix this problem a fit of the histogram curve will be performed and the analytical function obtained will be used to calculate a boundary value for every single image.

6.2.1 The Fit Function

Figure 6.3 shows that the function should tend to 0 both for $x \to 0$ and $x \to 255$ (images are in 8 bits greyscale format) and it should have a bump. This behaviour can be described by the function:

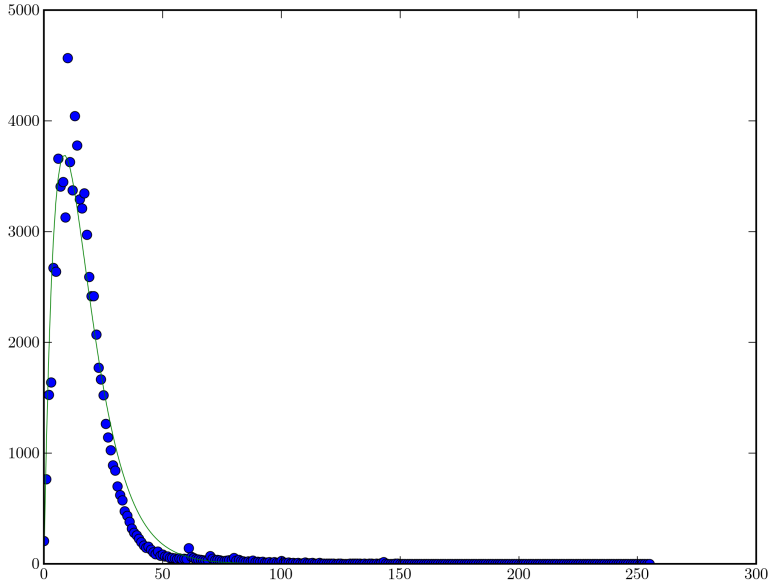

Figure 6.3: Distribution of regional maxima depending on their intensity

$$F(x) = ae^{-bx} - ae^{-cx} + d$$

where a, b, c, d are the parameters of the function. Since it is already known that the function should tend to 0, then d can already be considered 0 as well.

The least squares method is going to be used to fit the function. More robust criteria were also considered, but the used algorithm gave in all the simulations give satisfactory results.

The residuals will in the specific case look like:

$$r_i = m_i - F(i)$$

where m_i is the number of maxima having intensity i.

The parameters will be therefore found as:

$$\arg\min \sum_i r_i^2$$

6.2.2 Finding the Noise Level

Now that a, b, c, d are known (with d being 0) an analytical function approximating the distribution of regional maxima is available, as shown in Figure 6.3.

At this point a strategy to filter out maxima which are too faint is needed. Deciding where the noise level precisely lies is not an easy task but for this application an exact evaluation is not needed.

The solution adopted is to take note of the maximum of F and for which value i it happens:

$$\begin{aligned} i_m &= \arg\max F(i) \\ m_m &= F(i_m) \end{aligned}$$

Maxima below a certain intensity i_L need to be ignored to avoid the noise level. In order to find a proper i_L the following criterion is used:

$$F(i_L) = \frac{m_m}{\alpha}$$

which means that the maximum of F will be divided by α and all maxima whose intensity is lower than $F(i_L)$ will be ignored.

The choice of α is arbitrary but in this model a value of 100 has shown the best results.

6.3 Removing False Maxima

Every image captured by the C2 or C3 instrument contains its date and time. Most of these get removed in the image subtraction process, but data relative to the time remains visible and can be mistaken for a regional maxima, misleading the model. Figure 6.4 shows an example.

The solution adopted in the Matlab implementation simply ignored all the maxima in the lower left corner of the image. This is an easy and effective solution, but it does not allow to capture objects that may lie in between numbers.

To solve this a more sophisticated method has been developed for the Python version. This consists in considering all the regional maxima detected in the lower left corner of I_b with intensity equal to 255 (since this is the color used to write write the numbers on the picture) and for each of them checking if it lies on a number of the first image considered in the subtraction process.

6.3.1 Detecting Numbers

The detection of number is performed using a connected component analysis on the considered rectangle [3]. An 8-connectivity model is used and a set of objects are defined. These can be seen in Figure 6.5. At this point the size of each object is checked and, since the smallest object is a "dot" which is defined by 20 pixels, only objects with size greater than 20 are considered to be numbers.

Now a distinction between maxima belonging to a region defined by a number and stars is available and will be used to filter out bogus maxima.

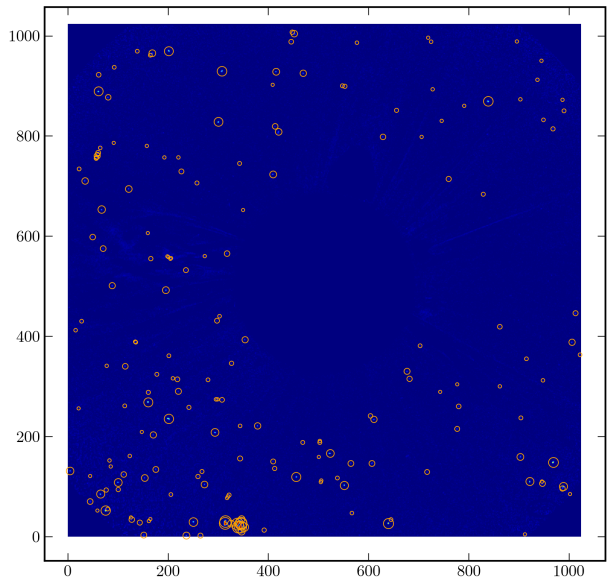

Figure 6.4: Object detection: many maxima are found in correspondence of the lower left corner because of the time-stamp

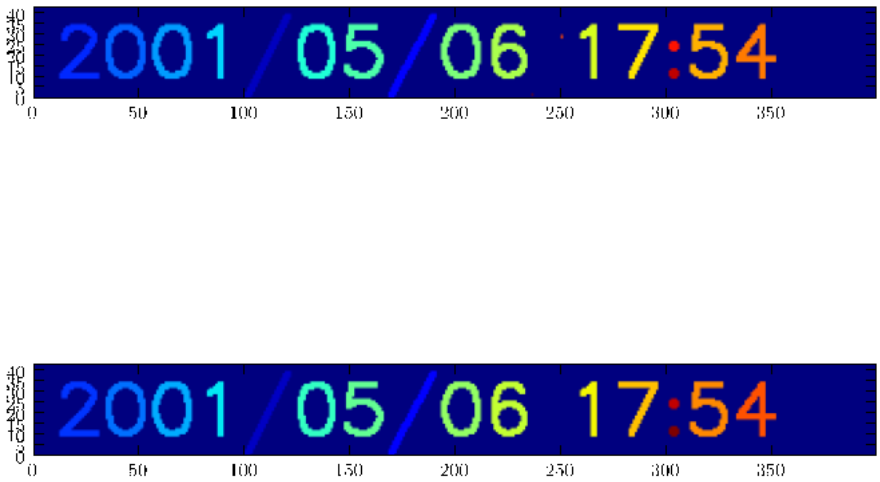

Figure 6.5: Connected component analysis: each color represents a different objects. The bottom image shows that the star object between numbers has been successfully detected and removed

Chapter 7

Computing the Properties of the Objects

Now that the regional maxima has been found and filtered, the position and intensity of every maxima is retrieved and saved. These were the only information used in the Matlab implementation of the model.

In the new model more information are calculated inspired by [1] in order to have a bigger amount of boundaries in the path tracing stage which is described in the next chapter.

A star will always appear in the image as a sharp circle, but comets can appear in many different ways. They can be sharp or blurred, they show have a tail or not, they can look as circles or ellipses. This makes the characterization of comets in images a difficult task, but these features will, at some extent, be constant over a sequence of images, so it is possible to actually use these characteristics to our advantage.

7.1 Sup-pixel Precision Center of the Object

If we consider a single object o whose coordinates are (i_o, j_o) and a small area A_o around the object (in this application over 5x5 pixels), then the center of the object can be defined as:

$$x_c = \frac{\sum_{A_o} I_b(i,j) \cdot i}{\sum_{A_o} I_b(i,j)}, \quad y_c = \frac{\sum_{A_o} I_b(i,j) \cdot j}{\sum_{A_o} I_b(i,j)}$$

This definition starts from the consideration that an object cannot be an isolated pixel for two reason:

1. the optics of the camera spreads point light source (as stars can be considered) over a set of pixels on the CCD sensor with a Gaussian distribution
2. I_b is in itself the result of the convolution over a Gaussian kernel

This also means that is an object appears as a Gaussian with a too narrow σ, then it is most likely a hot-pixel of the CCD sensor.

7.2 Estimated Maximum

The next step of the algorithm is the approximation of the observed intensity of the two-dimensional Gaussian profile given by:

$$G(i,j) = G_O e^{-\frac{\left[(i-x_c)^2+(j-y_c)^2\right]}{2h^2}} + b$$

where G_O is the estimated maximum of the object profile and b is the local sky background. The parameter h is the "width" of the typical object and its numerical value must be known a priori. In this application a value of 0.8 has been adopted and has been found experimentally.

To find G_O and b the least squares method will be used so the sums of the squared residuals will be minimized:

$$G_O, b = \arg\min \sum_{A_O} (G(i,j) - A_O(i,j))^2$$

7.3 Central Intensity

The central intensity I_0 can be estimated from the maximum intensity $I_{i_o j_o}$ by subtracting the local background intensity $B_{i_o j_o}$ from $I_{i_o j_o}$:

$$I_0 = I_{i_o j_o} - B_{i_o j_o}$$

where:

$$B_{i_o j_o} = \frac{\sum_{(i,j)\neq(i_o,j_o)} w_{ij} I_{ij}}{\sum_{(i,j)\neq(i_o,j_o)} w_{ij}}$$

w_{ij} are arbitrary weights. For this application they have been considered a constant.

7.4 Sharpness

The new parameter is defined and it is called and describes the ratio between the central intensity and the estimated maximum:

$$S = \frac{I_0}{G_O}$$

This practically gives an approximated evaluation of the fit of the Gaussian and, since this has a constant h, if the Gaussian didn't fit too good it means that the chosen h parameter was wrong. S can, in other words, be considered as a rough description of the sharpness of the object.

7.5 Elongation

Another way to describe an object can be observing its elongation and the direction. It can, in other words, be described as an ellipse with the major semi-axis at some angle to the x coordinate. This can be analyzed computing the description of the object by its second moments:

$$\begin{aligned} h_x^2 &= \frac{\sum_{A_O} I_{ij}(i-x_c)^2}{\sum_{A_O} I_{ij}} \\ h_x^2 &= \frac{\sum_{A_O} I_{ij}(j-y_c)^2}{\sum_{A_O} I_{ij}} \\ h_{xy} &= \frac{\sum_{A_O} I_{ij}(i-x_c)^2(j-y_c)^2}{\sum_{A_O} I_{ij}} \end{aligned}$$

Now the length of the semi-axes of the objects can be computed as an eigenvalues problem. Their length satisfy the equation:

$$\begin{vmatrix} \lambda - h_x & h_{xy} \\ h_{xy} & \lambda - h_y \end{vmatrix} = 0$$

Resolving this equation for λ gives the sizes of the semi-axes:

$$\lambda_{\pm} = \frac{1}{2}\left[(h_x + h_y) \pm \sqrt{(h_x + h_y)^2 + 4h_{xy}^2}\right]$$

The numerical values of it parameters are different of elongated objects, but approximately the same for, for instance, stars. With respect to their character the E parameter can be defined as:

$$E = 2\frac{\sqrt{(h_x + h_y)^2 + 4h_{xy}^2}}{h_x + h_y}$$

Chapter 8

Tracing the paths

At this point quite a few information for each objects are known:

- position in the image
- intensity in the image
- computed sub-pixel position
- computed intensity
- sharpness
- elongation

The choice of the first or the second two is arbitrary and does not affect the model in a sensitive way so, all in all, there are four parameters to start with.

8.1 Object Tracing vs Path Tracing

This is one of the major improvements since the first version of the model. The two approaches can be summarized as follows:

Object tracing Every object is checked in order to distinguish it from stars, cosmic noise etc. This means that if an object is mistakenly labeled as a star in one of the images of the sequence, then the whole path will not be recognized. This can cause some false negatives and happens when a comet and a star appear as a single object in one of the pictures.

Path tracing All the possible paths are computed and evaluated and no object is labelled. Each plausible path, formed by compatible objects, will finally be evaluated in order to find out if it is a star or a comet.

The path tracing technique is more demanding in terms of computation power and memory usage, but also makes it much easier to follow the path performed by one object in the sequence of images. On the other hand the object tracing technique only gave as a result the position of the candidate comet in the last image and the path had to be reconstructed backwards.

8.2 First level constraints

The first level of constraints rely on position, intensity, shape and elongation. This means that for each object in one image compatible objects in the following image will be searched. The constraint can be expressed as follows:

$$\begin{aligned} |x_{c_1} - x_{c_2}| &< d_x \\ |y_{c_1} - y_{c_2}| &< d_y \\ |I_1(x_{c_1}, y_{c_1}) - I_2(x_{c_2}, y_{c_2})| &< d_i \\ |S_1 - S_2| &< d_S \\ |E_1 - E_2| &< d_E \end{aligned}$$

which means that the difference between position, intensity, sharpness and elongation of objects in two consecutive images must be lower that a fixed parameter.

In addition we know that comets always move towards the sun so:

$$D_1 > D_2$$

where D represents the distance from the sun. This is easy to calculate since all the images have the center of the sun located at the center of the image.

At this stage the speed and direction of the so far plausible object couple are computed. They will be called respectively V and D.

8.3 Second level constraints

Now speeds and directions will be checked for each couple of images.

$$|V_{i,i+1} - V_{i+1,i+2}| < d_V$$
$$|D_{i,i+1} - D_{i+1,i+2}| < d_D$$

One of the main reasons of false positive at this point is that the corners of the images are very noisy and many false maxima are spotted. This means that many paths are spotted of hypothetical objects that move close to the corners and approaching the sun with an angle which is very close to 90 degrees. So adding another condition will get rid of these paths:

$$A_{1,2} < d_A$$

where A is the angle between the direction of the object and the segment that joins the object and the sun and d_A is considerably less than 90 degrees. It is interesting to notice that, in order to save computation power, this condition can only be applied once for the hole path (the beginning for instance, as supposed in the previous equation) since a condition for the stability of the direction is already present.

8.4 Filtering out stars

At this point the plausible paths contain both candidate comets and stars. Fortunately stars move in a very predictable way so it is easy to filter them out.

Stars slowly move in the images from left to right with a known speed and direction. This allows to formulate two conditions to detect stars:

$$V_{min} < V_i < V_{max}$$
$$D_{min} < D_i < D_{max}$$

Once stars are removed from the paths lists only candidate comets are left.

Chapter 9

Results

9.1 Implementation Notes

The whole model was implemented in Python, apart from the regional maxima algorithm was in C. A comparison in terms of speed performance between the current and the previous implementation in Matlab is very hard to perform since they ran on very different hardware. The Matlab implementation was developed on a Sun UNIX workstation while the Python one on a Compaq Personal Computer equipped with a 1.1 GHz Intel Celeron processor and 750 Mb of RAM. The CPU power of the Sun workstation is unknown but, since it is a shared server, it might be lower than the PC. On the other hand the RAM availability was incredibly higher on the UNIX system.

In both cases the algorithm was used to analyze the whole data-set from year 2006. In the first implementation this process took about one week while in the second it took about 36 hours. This cannot unfortunately be used to compare the speed of the two implementations or programming languages for the above stated reasons.

An important note to point out is that the 750Mb limit of the personal computer proved to be the main issue during the implementation since most of the speed optimizations were very demanding from the memory point of view. This took

to the decision to make the algorithm abort the analysis of an image sequence if more than 500 objects were found in the object detection phase. This measure has anyway also speed limit reasons since the computation path tracking phase is proportional to the square of the number or object to analyse. The main consequence of this forced limit is the inability to analyze C3 images since they in average contain about 1500 object above noise level.

The above mentioned simulation for the whole data-set of C2 images for year 2006 was taking advantage of some handy advanced features of Python. These include:

- HTTP connection (download the HTML file containing the links for the images, and download of the images)
- HTML parsing (to find the links to the images in the HTML file)
- Advanced data structures and Object Oriented programming (in particular lists and queues were used)
- log files handling
- email system (to send a message to the author whenever a possible comet was spotted)

These features would have been very hard or even impossible to get from Matlab.

9.2 Analysis of the 2006 data-set

The SOHO website kindly makes all the images from year 2006 easily available and they setup a web-page containing the links to all the pictures. All in all they are 13182 and the model has been launched to analyze all the sequences of four consecutive images.

Table 9.3 shows a synthesis of the results of two separate runs of the simulation: the first for $d_A = \frac{4\pi}{9}$ and the second for $d_A = \frac{\pi}{4}$. From now on the two simulations will be called S1 and S2.

As it can be easily noticed more than one third of sequences were not even taken into consideration because of the time lapse between single images being too big. Is these simulations the maximum allowed time lapse between two consecutive images was set to one hour.

Another sequence abortion criterion is the number of objects found for the reasons stated in the previous section. More than one sixth of the sequences was not completed because one of the images in it contained more than 500 objects.

In S2, as obvious from the tighter condition, the possible comets found are less than in S1. And all of the candidates found are actually comets. On the other side some false negatives are also found in S1 while they are shown in S2. Then again S2 contains several false positives. This shows that the choice of d_A is a very delicate one and it can be determinant in the identification of a comet.

Some examples of detected comets can be seen in Figures 9.1-9.4.

Figure 9.1 shows a very bright comet being detected and tracked in a sequence of images. This is an easy object to detect since it is extremely well defined from the background and it has sharp edges. One interesting observation can be done about the second image where a close look can tell that an object appears right below the comet making it appear as a tail. The algorithm was able to keep track of the comet because of the loose condition on the shape parameter.

Figure 9.2 shows another comet, not as bright as the previous, but still easy to detect. This object shows indeed a faint short tail, which is not clearly visible in the first image, but is evident in the following ones. It can be noticed that in the sequence some object of comparable size and intensity appear. The algorithm is able to sort things about thanks to the conditions about speed and direction.

Figure 9.3 shows a very interesting comet. Opposite to the previous examples this object appears to be very faint and blurry. Luckily all of the 4 object's intensities were right above the noise level.

Figure 9.4 shows how the algorithm works also when the comet shows a very bright tail as in this case.

Figure 9.5 shows what happens when there is no condition about angle between the direction of the object and the segment that join the object to the sun. It can be seen the the boundaries of the images are very noisy, probably due to JPEG compression, and this leads to the identification of many spurious regional maxima. The big amount of maxima detected increases the probability that a sequence of compatible objects may appear at the right time in the right spot of the images.

	number	%
total simulations	13179	100.00
aborted for time lapse	4729	35.88
aborted for number of objects	2011	15.26
possible comets S1	48	0.36
possible comets S2	19	0.14

Table 9.1: Summary of the simulations results for the 2006 data-set

9.3 Possible future developments

The comment about Figure 9.5 paves the way to a more important one. If the computation power and memory amount allowed to consider all the detected regional maxima, without avoiding the noise level, would this model still be valid and efficient or would it crash against the probability of coincidences that "look" like a proper comet? Of course the answer to this question cannot be delivered yet, but a consideration can be done.

There will clearly be some problems since it can be easily predicted that coincidences like the one shown in Figure 9.5 would appear very frequently. Possible solutions to this problem would be to tighten the current conditions and, more importantly, finding and defining more criteria to characterise the objects in order to increase the number of conditions.

Another future improvement might be to automatically calculate the orbit of every single found comet in order to be able to look if some pattern can be found. This is very interesting especially from the astronomical point of view, so important that actually it is already being done with all the SOHO comets that are being found. This led to the definition of comet groups, where Kreuz is the most common one.

Possible further developments in order to increase execution speed and optimize memory management are also of course possible.

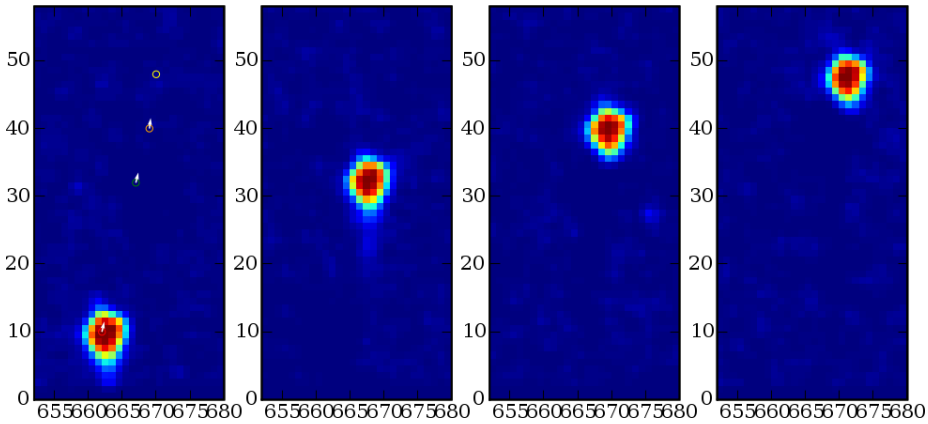

Figure 9.1: A very bright comet as it enters the field of view of the camera.

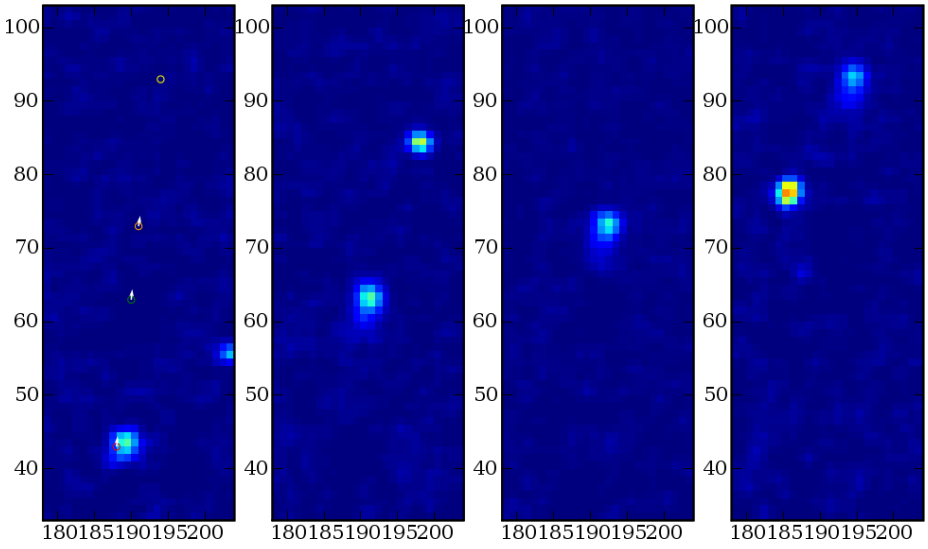

Figure 9.2: A bright comet as it enters the field of view of the camera.

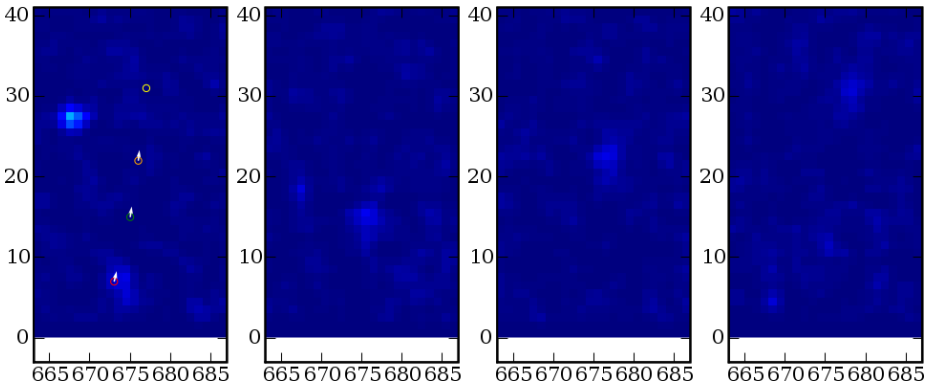

Figure 9.3: A blurry faint comet as it enters the field of view of the camera.

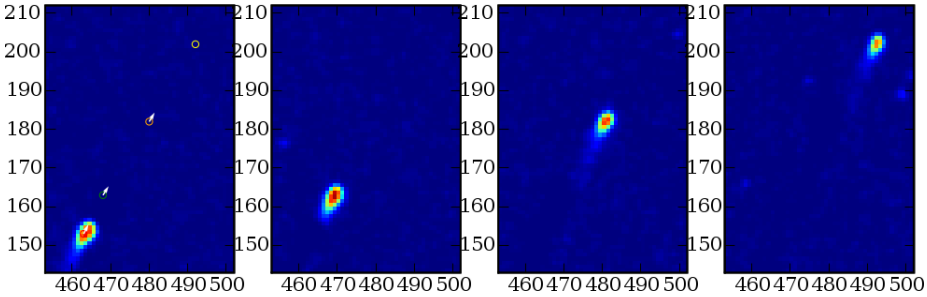

Figure 9.4: A bright tailed comet as it enters the field of view of the camera.

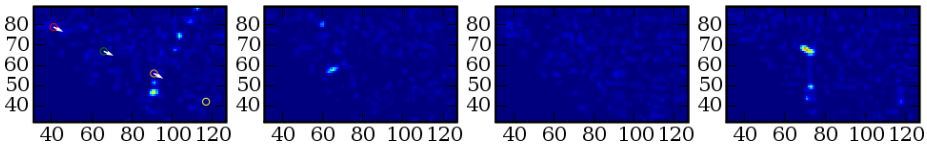

Figure 9.5: A false positive object.

Chapter 10

Conclusion

The results of this project proved that image analysis can be a useful and sometimes invaluably precious tool to find a solution to problem otherwise solved with hours of work.

The model proved to be efficient, fast and precise in the analysed cases and, even if not yet perfect, it's capability to run unattended makes it ready to be used to aid astronomers to detect and track comets in the images provided by the instruments of the SOHO satellite.

Many mathematical tools were used to achieve the result: mathematical morphology ([5], [6], [7]), connected component analysis, optimization and data fitting. These were glued together in the Python inplementation which proved to be a mature and versatile tool for scientific computing.

The aim mentioned in the introduction seems to be reached and a small grain of sand can be added to the pile.

Appendix A

Source Code

```
from numpy import *
from scipy import ndimage
from scipy.optimize import leastsq, fminbound, anneal
from scipy.optimize.minpack import fsolve
from scipy.weave import inline
from scipy.weave import converters
import Image
import pylab
import logging
import re
import urllib
import sys
import time
import os
from HTMLParser import HTMLParser
import smtplib
from email.mime.image import MIMEImage
from email.mime.multipart import MIMEMultipart

def grey_reconstruct(start, mask, fp):
    """Returns the greyscale reconstruction by dilation
        of start with footprint fp"""
    dilated = start # initial step
```

```
    while True: # change since last iteration?
        new_dilated = ndimage.grey_dilation(dilated,
            footprint=fp)
        indices = where(new_dilated > mask)
        new_dilated[indices] = mask[indices]
        if alltrue(new_dilated == dilated):
            break
        else:
            dilated = new_dilated
    return new_dilated

def fast_grey_reconstruct(start, mask):
        marker = start
        height, width = start.shape
        code = """
  int ix,iy,ox,oy,offset;
  int currentQ,currentP;
  int pixPerImg=width*height;
  int val1,val2,maxVal=0;
  int *istart,*irev,*ifwd;

  for(offset=pixPerImg-1;offset>=0;offset--)
    if(marker[offset]>maxVal)
      maxVal=marker[offset];
  istart=(int*)malloc((maxVal+pixPerImg*2)*sizeof(int))
    ;
  irev=istart+maxVal;
  ifwd=irev+pixPerImg;
  for(offset=-maxVal;offset<0;offset++)
    irev[offset]=offset;
  for(offset=pixPerImg-1;offset>=0;offset--){
    if(marker[offset]>0){
      val1=-marker[offset];
      irev[offset]=val1;
      ifwd[offset]=irev[val1];
      irev[val1]=offset;
      if(ifwd[offset]>=0)
        irev[ifwd[offset]]=offset;
    }
  }
  for(currentQ=-maxVal;currentQ<0;currentQ++){
    currentP=irev[currentQ];
    while(currentP>=0){
      irev[currentQ]=ifwd[currentP];
```

```
      irev[currentP] = currentQ;
      ix = currentP%width;
      iy = currentP/width;
      for (oy = iy-1; oy <= iy+1; oy++) {
        for (ox = ix-1; ox <= ix+1; ox++) {
          if (ox >= 0 && oy >= 0 && ox < width && oy <
              height) {
            offset = ox+oy*width;
            val1 = marker[offset];
            val2 = marker[currentP]<mask[offset]?marker[
              currentP]:mask[offset];
            if (val1 < val2) {
              if (val1 != 0) {
                ifwd[irev[offset]] = ifwd[offset];
                if (ifwd[offset] >= 0)
                  irev[ifwd[offset]] = irev[offset];
              }
              marker[offset] = val2;
              irev[offset] = -val2;
              ifwd[offset] = irev[-val2];
              irev[-val2] = offset;
              if (ifwd[offset] >= 0)
                irev[ifwd[offset]] = offset;
            }
          }
        }
      }
      currentP = irev[currentQ];
    }
  }
  free(istart);
  return_val = marker;
  """
        inline(code, ['marker', 'mask', 'height', 'width'
            ], verbose=1)
        return marker

def cart2pol(x,y):
    r = sqrt(x**2+y**2)
    phi = arccos(x/r)
    if y < 0:
        phi = 2*pi-phi
    return phi, r
```

```
def angle_diff(a1,a2):
    b = a2-a1
    if b <= -pi:
        b = b+2*pi
    elif b > pi:
        b = b-2*pi
    return b

def residuals(p, y, x):
    a, b, c = p
    err = y-(-a*exp(-b*x)+a*exp(-c*x))
    return err

def func_eval(x, p):
    a, b, c, d = p
    f = -(-a*exp(-b*x)+a*exp(-c*x)-d)
    return f

def load_image(name):
    tmp = array(Image.open(name).getdata())
    tmp = tmp.reshape(1024,1024)
    #tmp = tmp[:,:-100]
    return tmp

def find_maxima(im):
    #reg_maxima = im - grey_reconstruct(im-1, im, ones
        ([3,3]))
    gr = fast_grey_reconstruct(im-1, im)
    reg_maxima = im - gr
    return reg_maxima

def detect_objects(orig2, blurred, ratio):
    maxima = find_maxima(blurred) # find regional maxima

    maxima_sum = maxima.sum() # to output

    img = blurred[where(maxima>0)]

    x = arange(256.)

    (n, bins) = histogram(img, x) # divide the regional
        maxime in bins according to their intensity

    p0 = [10000., 0.5, 0.5] # starting point
```

```
p, tmp = leastsq(residuals, p0, args=(n, x), maxfev
    =2000) # find the curve that fits the distribution
#p, tmp = leastsq(lambda p:residuals(p,n,x), p0,
    maxfev=2000) # find the curve that fits the
    distribution

#print p
new_p = concatenate((p, [0.]))

#print new_p
x_max = fminbound(func_eval, 0, 255, args=(new_p,)) #
    find the X for the max
#x_max = fminbound(lambda x:func_eval(x,new_p), 0,
    255) # find the X for the max

y_max = -func_eval(x_max, new_p) # evaluate the max

#y = -p[0]*exp(-p[1]*x)+p[0]*exp(-p[2]*x)

new_p = concatenate((p, [y_max*ratio]))

x_stop = fsolve(func_eval, array([x_max*2]), (new_p,)
    )
#print "X stop coordinate:", x_stop
final_maxima = maxima

final_maxima[where(blurred<x_stop)] = 0


# remove maxima on numbers
# ------------------------

# make a list of maxima in the bottom left rectangle
bl_maxima = final_maxima[980:1023, 0:399] # cut the
    interesting part of the maxima matrix
bl_img = orig2[980:1023, 0:399] # do the same for the
    original image
bl_img[bl_img<255] = 0 # only interested in the date
bl_img[bl_img==255] = 1 # date is in pure white

bl_l, bl_n = ndimage.label(bl_img) # labels

if bl_n > 0:
    bl_objects = ndimage.find_objects(bl_l) #
```

```
        objectsrelated to labels
    bl_obj_array = array(bl_objects) # to array

    sizes = ndimage.sum(bl_img, labels=bl_l, index=
        range(1,bl_n+1)) #sizes of the objects
    sizes = array(sizes) # to array
    bl_obj_indices = where(sizes>21) # minimum size =
         20 pixels

    for i in bl_obj_indices[0]: # if size is less
        than that...
            #bl_l[bl_objects[i]] = 0 # label of that
                obj = 0
            bl_maxima[bl_objects[i]] = 0

    #bl_maxima[bl_l] = 0

    final_maxima[980:1023, 0:399] = bl_maxima

l, n = ndimage.label(final_maxima, structure=ones
    ([3,3]))
objects = ndimage.find_objects(l)

objects_size = ndimage.sum(final_maxima, labels=l,
    index=range(1,n+1))
final_maxima[l[objects_size>1]] = 0

l, n = ndimage.label(final_maxima, structure=ones
    ([3,3]))
objects = ndimage.find_objects(l)

objects_list = ndimage.center_of_mass(final_maxima,
    labels=l, index=range(1,n+1))
objects = array(objects_list)

obj_y = objects[:,0]
obj_x = objects[:,1]

obj_v = []
for i in objects_list:
    obj_v.append(blurred[i[0], i[1]])

obj_v = array(obj_v)
```

```
    return [obj_x, obj_y, obj_v, x_stop, maxima_sum]

def gaussian(x, y, p, x_c, y_c, h):
    return p[0]*exp(-(((x_c-x)/h)**2+((y_c-y)/h)**2)/2) +
        p[1]

def errorfunction(p, data, I, J, x_c, y_c, h):
    return ravel(gaussian(I, J, p, x_c, y_c, h) - data)

def distance_from_sun(x, y):
        return sqrt((x-512)**2+(y-512)**2)

def distance(x1, y1, x2, y2):
        return sqrt((x1-x2)**2+(y1-y2)**2)

def get_objects_info(image, size, obj_x, obj_y, obj_v, h)
    :
    data = []
    for o in range(len(obj_x)):
        # cut the rectangle of interesting data
        x1 = max(floor(obj_x[o])-size, 0)
        x2 = min(ceil(obj_x[o])+size+1, 1023)
        y1 = max(floor(obj_y[o])-size, 0)
        y2 = min(ceil(obj_y[o])+size+1, 1023)
        img = image[y1:y2,x1:x2]
        I, J = mgrid[y1:y2,x1:x2]
        imgsum = img.sum()
        x_c = sum(img*I)/imgsum
        y_c = sum(img*J)/imgsum

        params = [obj_v[o], img.min()]
        p, success = leastsq(errorfunction, params, args
            =(img, I, J, x_c, y_c, h), maxfev=5000)
        G_0, B = p

        I_0 = obj_v[o] - B

        sharp = I_0/G_0
        h_x  = sqrt(sum(img*(I-x_c)**2)/imgsum)
        h_y  = sqrt(sum(img*(J-y_c)**2)/imgsum)
        h_xy = sum(img*(I-x_c)*(J-y_c))/imgsum
        shape = 2*sqrt((h_x-h_y)**2+4*h_xy**2)/(h_x+h_y)

        data.append([x_c, y_c, G_0, I_0, sharp, h_x, h_y,
```

```
            h_xy, shape, distance_from_sun(x_c, y_c)])

    return data

def find_path2(data1, data2, data3, data4, td1, td2, td3)
    :
        data1=array(data1)
        data2=array(data2)
        data3=array(data3)
        data4=array(data4)

        lvl1_tolerance = array([30, 30, 60, 60, 0.5, 0.5,
            0.5, 0.5, 0.5, 30])
        d1 = int16(data1[newaxis, :, :])
        d2 = int16(data2[:, newaxis, :])
        close_enough1 = less(absolute(d2-d1),
            lvl1_tolerance).all(axis=2)
        close_enough2 = less(d2[:,:,-1]+4, d1[:,:,-1])
        #print d2[:,:,0]-d1[:,:,0] , d2[:,:,1]-d1[:,:,1]
        #close_enough2 = less(anglediff(cart2pol(d2
            [:,:,0]-d1[:,:,0],d2[:,:,1]-d1[:,:,1])[0],
            cart2pol(512-d1[:,:,0],512-d1[:,:,1])[0]),
            3.14/4)
        close_enough  = close_enough1 & close_enough2
        d1_d2 = int16(array(where(close_enough)))
        del d1, d2

        #d2_idx = where(close_enough.any(axis=1))[0]
        #data2 = array(data2)[d2_idx,:]
        d2 = int16(data2[newaxis, :, :])
        d3 = int16(data3[:, newaxis, :])
        close_enough1 = less(absolute(d3-d2),
            lvl1_tolerance).all(axis=2)
        close_enough2 = less(d3[:,:,-1]+4, d2[:,:,-1])
        close_enough  = close_enough1 & close_enough2
        d2_d3 = int16(array(where(close_enough)))
        del d2, d3

        #d3_idx = where(close_enough.any(axis=1))[0]
        #data3 = array(data3)[d3_idx,:]
        d3 = int16(data3[newaxis, :, :])
        d4 = int16(data4[:, newaxis, :])
        close_enough1 = less(absolute(d4-d3),
            lvl1_tolerance).all(axis=2)
```

```
        close_enough2 = less(d4[:,:,-1]+4, d3[:,:,-1])
        close_enough  = close_enough1 & close_enough2
        d3_d4 = int16(array(where(close_enough)))
        del d3, d4

        del close_enough1, close_enough2

#       paths = []
#       for i in range(d1_d2.shape[1]):
#               for j in range(d2_d3.shape[1]):
#                       for k in range(d3_d4.shape[1]):
#                               if d1_d2[0,i] == d2_d3[1,
    j] and d2_d3[0,j] == d3_d4[1,k]:
#                                       paths.append([
    d1_d2[1,i], d1_d2[0,i], d2_d3[0,j], d3_d4[0,k]])

        if d1_d2.shape[1]*d2_d3.shape[1]*d3_d4.shape[1] >
            2000000:
                return array([[-1]], ndmin=2), array
                    ([[-1]], ndmin=2)

        p1 = int16(d1_d2.repeat(d2_d3.shape[1]*d3_d4.
            shape[1], axis=1))
        p2 = int16(tile(d2_d3.repeat(d3_d4.shape[1], axis
            =1), d1_d2.shape[1]))
        p3 = int16(tile(d3_d4, d1_d2.shape[1]*d2_d3.shape
            [1]))
        p = concatenate((p1,p2,p3))
        del p1, p2, p3
        idx = intersect1d(array(where(p[0,:] == p[3,:]))
            [0,:], array(where(p[2,:] == p[5,:]))[0,:])
        goodpath = p[:,idx]
        paths = goodpath[[1,0,2,4], :].T.tolist()

        paths2 = []
        coords = []
        for i in paths:
            x1 = data1[i[0],1]
            y1 = data1[i[0],0]
            x2 = data2[i[1],1]
            y2 = data2[i[1],0]
            x3 = data3[i[2],1]
            y3 = data3[i[2],0]
            x4 = data4[i[3],1]
```

```
                y4 = data4[i[3],0]

                d1, r1 = cart2pol(x2-x1,y2-y1)
                d2, r2 = cart2pol(x3-x2,y3-y2)
                d3, r3 = cart2pol(x4-x3,y4-y3)

                v1 = r1/td1
                v2 = r2/td2
                v3 = r3/td3

                angles = absolute(array([anglediff(d2,d1),
                    anglediff(d3,d2)]))

                #if less(angles, pi/16).all() and less(array
                    ([v2-v1, v3-v2]), 0.1).all() and greater(
                    absolute(array([d1,d2,d3])-6.15), 0.1).all
                    () and greater(array([v1,v2,v3]), 0.5).all
                    () and less(anglediff(cart2pol(x2-x1,y2-y1
                    )[0],cart2pol(512-x1,512-y1)[0]), 3.14/4):
                if less(angles, pi/16).all() and less(array([
                    v2-v1, v3-v2]), 0.1).all() and greater(
                    absolute(array([d1,d2,d3])-6.15), 0.1).all
                    () and greater(array([v1,v2,v3]), 0.5).all
                    () and less( abs(arccos(abs(y1-512)/
                    distance_from_sun(x1, y1)-arccos(abs(y2-y1
                    )/distance(x1, y1, x2, y2)))), 1.22):
                        paths2.append(i)
                        coords.append([x1, x2, x3, x4, y1, y2
                            , y3, y4])

        c = int16(array(coords, ndmin=2))
        c[:,4:] = 1023 - c[:,4:]

        return c, paths2

def minutes_in_day(filename):
        hour   = int(filename[-11:-9])
        minute = int(filename[-9:-7])
        return hour*60+minute

def timediff(t1, t2):
        return (t2-t1)%1440
```

```
def find_comets(filename1, filename2, filename3,
    filename4):
    sigma = 0.7 # check also 0.6 and 0.7
    h = 0.8 # found experimentally

    t1 = minutes_in_day(filename1)
    t2 = minutes_in_day(filename2)
    t3 = minutes_in_day(filename3)
    t4 = minutes_in_day(filename4)

    timediff1 = timediff(t1,t2)
    timediff2 = timediff(t2,t3)
    timediff3 = timediff(t3,t4)

    if max([timediff1, timediff2, timediff3]) > 60:
            logging.warning('timelapse between images is
                too big. ['+filename1+', '+filename2+', '+
                filename3+', '+filename4+'] aborting.')
            return -1

    logging.info('loading image '+filename1)
    orig1 = load_image(filename1)
    logging.info('loading image '+filename2)
    orig2 = load_image(filename2)
    logging.info('loading image '+filename3)
    orig3 = load_image(filename3)
    logging.info('loading image '+filename4)
    orig4 = load_image(filename4)

    # ----------------
    # IMAGE CLEANING
    # ----------------
    logging.info("starting images subtractions")
    im1 = orig1 - orig2
    im1[im1<0] = 0

    im2 = orig2 - orig1
    im2[im2<0] = 0

    im3 = orig3 - orig2
    im3[im3<0] = 0

    im4 = orig4 - orig3
    im4[im4<0] = 0
```

```
# ----------------
# OBJECT FINDING
# ----------------
logging.info("starting image blurring with sigma="+
    str(sigma))
blurred1 = ndimage.gaussian_filter(im1, sigma) # blur
     with given sigma
blurred2 = ndimage.gaussian_filter(im2, sigma) # blur
     with given sigma
blurred3 = ndimage.gaussian_filter(im3, sigma) # blur
     with given sigma
blurred4 = ndimage.gaussian_filter(im4, sigma) # blur
     with given sigma

logging.info("starting objects detection")
obj_x1, obj_y1, obj_v1, x_stop1, maxima_sum1 =
    detect_objects(orig1, blurred1, 0.02)
obj_x2, obj_y2, obj_v2, x_stop2, maxima_sum2 =
    detect_objects(orig2, blurred2, 0.02)
obj_x3, obj_y3, obj_v3, x_stop3, maxima_sum3 =
    detect_objects(orig3, blurred3, 0.02)
obj_x4, obj_y4, obj_v4, x_stop4, maxima_sum4 =
    detect_objects(orig4, blurred4, 0.02)

del orig1, orig2, orig3, orig4

logging.info("objects found in image1: "+str(len(
    obj_x1)))
logging.info("objects found in image2: "+str(len(
    obj_x2)))
logging.info("objects found in image3: "+str(len(
    obj_x3)))
logging.info("objects found in image4: "+str(len(
    obj_x4)))

if max([len(obj_x1), len(obj_x2), len(obj_x3), len(
    obj_x4)]) > 500:
        logging.warning('too many objects detected. 
            aborting.')
        return -2

# pylab.scatter(obj_x, 1023-obj_y, alpha=0)
# pylab.show()
```

```
# ------------------------
# FIND OBJECTS PROPERTIES
# ------------------------
logging.info("starting objects parameters detection")
data1 = get_objects_info(blurred1, 2, obj_x1, obj_y1,
    obj_v1, h)
data2 = get_objects_info(blurred2, 2, obj_x2, obj_y2,
    obj_v2, h)
data3 = get_objects_info(blurred3, 2, obj_x3, obj_y3,
    obj_v3, h)
data4 = get_objects_info(blurred4, 2, obj_x4, obj_y4,
    obj_v4, h)



# -----------------
# FIND COMETS PATH
# -----------------
logging.info("starting paths detection")
c, paths = find_path2(data1, data2, data3, data4,
   timediff1, timediff2, timediff3)
if len(paths)>0 and c[0,0] == -1:
        logging.info('too many objects after level 1.
            aborting.')
        return -3
counter = 1
if len(paths)>0:
        for i in range(c.shape[0]):
            row = c[i,:]
            logging.info('possible path: [x1 x2 x3 x4
                y1 y2 y3 y4]')
            logging.info(str(row))
            pylab.clf()
            pylab.subplot(141)
            pylab.imshow(blurred1, interpolation='
                nearest')
            pylab.scatter([row[0]], [row[4]], alpha
                =0, edgecolor='red')
            pylab.scatter([row[1]], [row[5]], alpha
                =0, edgecolor='green')
            pylab.scatter([row[2]], [row[6]], alpha
                =0, edgecolor='orange')
            pylab.scatter([row[3]], [row[7]], alpha
```

```
    =0, edgecolor='yellow')
pylab.quiver([row[0]], [row[4]], [row[1]-
    row[0]], [row[5]-row[4]], color='white
    ')
pylab.quiver([row[1]], [row[5]], [row[2]-
    row[1]], [row[6]-row[5]], color='white
    ')
pylab.quiver([row[2]], [row[6]], [row[3]-
    row[2]], [row[7]-row[6]], color='white
    ')
pylab.axis([row[0:4].min()-10, row[0:4].
    max()+10, row[4:].min()-10, row[4:].
    max()+10])
pylab.subplot(142)
pylab.imshow(blurred2, interpolation='
    nearest')
pylab.axis([row[0:4].min()-10, row[0:4].
    max()+10, row[4:].min()-10, row[4:].
    max()+10])
pylab.subplot(143)
pylab.imshow(blurred3, interpolation='
    nearest')
pylab.axis([row[0:4].min()-10, row[0:4].
    max()+10, row[4:].min()-10, row[4:].
    max()+10])
pylab.subplot(144)
pylab.imshow(blurred4, interpolation='
    nearest')
pylab.axis([row[0:4].min()-10, row[0:4].
    max()+10, row[4:].min()-10, row[4:].
    max()+10])
pylab.savefig(filename1[:-4]+'_'+str(
    counter)+'.png')
pylab.savefig(filename1[:-4]+'_'+str(
    counter)+'.pdf', dpi=300)

# Create the container (outer) email
    message.
msg = MIMEMultipart()
msg['Subject'] = 'Possible Comet in '+
    filename1[:-4]
# me == the sender's email address
# family = the list of all recipients'
    email addresses
```

```
                msg['From'] = 'emanuelez'
                msg['To'] = 'emanuelez@gmail.com'
                msg.preamble = 'Possible comet found in '
                    +filename1[:-4]+ 'with coordinates: '+
                      str(row)
                fp = open(filename1[:-4]+'_'+str(counter)
                    +'.png', 'rb')
                img = MIMEImage(fp.read())
                fp.close()
                msg.attach(img)
                s = smtplib.SMTP('localhost')
                s.sendmail('emanuelez', 'emanuelez@gmail.
                    com', msg.as_string())
                s.close()

                counter = counter+1
    return 1

class MyHTMLParser(HTMLParser):

    imgList = []
    todoList = []
    start = 24950

    def reload(self):
        try:
            f = urllib.urlopen('http://sohowww.nascom.
                nasa.gov/data/realtime/realtime-c2-1024-
                all.html')
            self.imgList = []
            self.todoList = []
            self.feed(f.read())
            self.todoList.sort()
            #self.todoList.reverse
        except IOError:
            sys.stderr.write('ERROR: cannot open URL\n')
        except HTMLParseError:
            sys.stderr.write('ERROR: HTML parsing failed\
                n')

    def handle_starttag(self, tag, attrs):
        if tag == 'a':
            newstr = str(attrs[0][1])
            if re.search('gif', newstr) != None:
```

```
                if (newstr in self.imgList) == False:
                    self.todoList.append(newstr)

def main():
    logging.basicConfig(level=logging.DEBUG,
                        format='%(asctime)s %(levelname)s %(
                            message)s',
                        filename='/tmp/comets.log',
                        filemode='w')
    logging.info('initiated image sequence')

    current = []
    newest = ""
    filename1 = ""
    ch = MyHTMLParser()
    while True:
        ch.reload()
        if len(ch.todoList)-ch.start > 3:
                current = []

                path = ch.todoList[ch.start]
                filename = path[-20:]
                current.append(filename)
                urllib.urlretrieve('http://sohowww.nascom
                    .nasa.gov'+path, filename)

                path = ch.todoList[ch.start+1]
                filename = path[-20:]
                current.append(filename)
                urllib.urlretrieve('http://sohowww.nascom
                    .nasa.gov'+path, filename)

                path = ch.todoList[ch.start+2]
                filename = path[-20:]
                current.append(filename)
                urllib.urlretrieve('http://sohowww.nascom
                    .nasa.gov'+path, filename)

                path = ch.todoList[ch.start+3]
                filename = path[-20:]
                current.append(filename)
                urllib.urlretrieve('http://sohowww.nascom
                    .nasa.gov'+path, filename)
```

```
                find_comets(current[0], current[1],
                    current[2], current[3])

                ch.start = ch.start+1

if __name__ == "__main__":
  main()
```

Appendix B

Log Files

Here part of the log files is inserted. Only the relevant parts and are displayed and they show all the times a candidate comet is found.

B.1 S1

```
2007-09-17 14:41:35,654 INFO starting objects detection
2007-09-17 14:41:39,899 INFO objects found in image1: 377
2007-09-17 14:41:39,900 INFO objects found in image2: 306
2007-09-17 14:41:39,900 INFO objects found in image3: 356
2007-09-17 14:41:39,900 INFO objects found in image4: 395
2007-09-17 14:41:39,900 INFO starting objects parameters detection
2007-09-17 14:41:43,412 INFO starting paths detection
2007-09-17 14:41:45,214 INFO loading image 20060104_0954_c2.gif
2007-09-17 14:41:45,558 INFO loading image 20060104_1006_c2.gif
2007-09-17 14:41:45,907 INFO loading image 20060104_1030_c2.gif
2007-09-17 14:41:46,255 INFO loading image 20060104_1054_c2.gif
2007-09-17 14:41:46,602 INFO starting images subtractions
2007-09-17 14:41:47,054 INFO starting image blurring with sigma=0.7
2007-09-17 14:41:48,283 INFO starting objects detection
```

```
2007-09-17 14:41:52,408 INFO objects found in image1: 363
2007-09-17 14:41:52,409 INFO objects found in image2: 356
2007-09-17 14:41:52,409 INFO objects found in image3: 395
2007-09-17 14:41:52,409 INFO objects found in image4: 379
2007-09-17 14:41:52,409 INFO starting objects parameters detection
2007-09-17 14:41:56,083 INFO starting paths detection
2007-09-17 14:41:56,527 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 14:41:56,528 INFO [ 926  929  929  930 1014  985  971  956]
2007-09-17 14:42:06,202 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 14:42:06,203 INFO [985 987 994 996 964 948 923 901]
--
2007-09-17 15:21:44,278 INFO starting objects detection
2007-09-17 15:21:48,382 INFO objects found in image1: 343
2007-09-17 15:21:48,383 INFO objects found in image2: 391
2007-09-17 15:21:48,383 INFO objects found in image3: 364
2007-09-17 15:21:48,383 INFO objects found in image4: 363
2007-09-17 15:21:48,383 INFO starting objects parameters detection
2007-09-17 15:21:52,034 INFO starting paths detection
2007-09-17 15:21:53,819 INFO loading image 20060111_1154_c2.gif
2007-09-17 15:21:54,183 INFO loading image 20060111_1206_c2.gif
2007-09-17 15:21:54,554 INFO loading image 20060111_1230_c2.gif
2007-09-17 15:21:54,919 INFO loading image 20060111_1254_c2.gif
2007-09-17 15:21:55,292 INFO starting images subtractions
2007-09-17 15:21:55,733 INFO starting image blurring with sigma=0.7
2007-09-17 15:21:56,843 INFO starting objects detection
2007-09-17 15:22:01,048 INFO objects found in image1: 387
2007-09-17 15:22:01,048 INFO objects found in image2: 364
2007-09-17 15:22:01,049 INFO objects found in image3: 363
2007-09-17 15:22:01,049 INFO objects found in image4: 405
2007-09-17 15:22:01,049 INFO starting objects parameters detection
2007-09-17 15:22:04,897 INFO starting paths detection
2007-09-17 15:22:05,450 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 15:22:05,451 INFO [ 954  957  962  965 1008  995  977  957]
--
2007-09-17 16:27:26,802 INFO starting objects detection
2007-09-17 16:27:30,961 INFO objects found in image1: 312
2007-09-17 16:27:30,962 INFO objects found in image2: 332
2007-09-17 16:27:30,962 INFO objects found in image3: 311
2007-09-17 16:27:30,962 INFO objects found in image4: 319
2007-09-17 16:27:30,962 INFO starting objects parameters detection
2007-09-17 16:27:34,270 INFO starting paths detection
2007-09-17 16:27:35,891 INFO loading image 20060124_1054_c2.gif
2007-09-17 16:27:36,237 INFO loading image 20060124_1106_c2.gif
2007-09-17 16:27:36,586 INFO loading image 20060124_1130_c2.gif
```

```
2007-09-17 16:27:36,999 INFO loading image 20060124_1154_c2.gif
2007-09-17 16:27:37,349 INFO starting images subtractions
2007-09-17 16:27:37,796 INFO starting image blurring with sigma=0.7
2007-09-17 16:27:38,956 INFO starting objects detection
2007-09-17 16:27:43,108 INFO objects found in image1: 327
2007-09-17 16:27:43,108 INFO objects found in image2: 311
2007-09-17 16:27:43,109 INFO objects found in image3: 319
2007-09-17 16:27:43,109 INFO objects found in image4: 300
2007-09-17 16:27:43,109 INFO starting objects parameters detection
2007-09-17 16:27:46,256 INFO starting paths detection
2007-09-17 16:27:46,506 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 16:27:46,507 INFO [ 921  917  913  905 1004  994  973  951]
--
2007-09-17 18:38:42,612 INFO starting objects detection
2007-09-17 18:38:46,843 INFO objects found in image1: 326
2007-09-17 18:38:46,844 INFO objects found in image2: 296
2007-09-17 18:38:46,844 INFO objects found in image3: 343
2007-09-17 18:38:46,844 INFO objects found in image4: 343
2007-09-17 18:38:46,844 INFO starting objects parameters detection
2007-09-17 18:38:50,295 INFO starting paths detection
2007-09-17 18:38:51,883 INFO loading image 20060221_1454_c2.gif
2007-09-17 18:38:52,274 INFO loading image 20060221_1506_c2.gif
2007-09-17 18:38:52,616 INFO loading image 20060221_1530_c2.gif
2007-09-17 18:38:52,978 INFO loading image 20060221_1554_c2.gif
2007-09-17 18:38:53,326 INFO starting images subtractions
2007-09-17 18:38:53,791 INFO starting image blurring with sigma=0.7
2007-09-17 18:38:54,935 INFO starting objects detection
2007-09-17 18:38:59,271 INFO objects found in image1: 312
2007-09-17 18:38:59,271 INFO objects found in image2: 343
2007-09-17 18:38:59,272 INFO objects found in image3: 343
2007-09-17 18:38:59,272 INFO objects found in image4: 374
2007-09-17 18:38:59,272 INFO starting objects parameters detection
2007-09-17 18:39:02,965 INFO starting paths detection
2007-09-17 18:39:03,241 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 18:39:03,242 INFO [967 968 969 968 916 897 874 859]
--
2007-09-17 22:09:38,704 INFO starting objects detection
2007-09-17 22:09:42,985 INFO objects found in image1: 333
2007-09-17 22:09:42,985 INFO objects found in image2: 262
2007-09-17 22:09:42,985 INFO objects found in image3: 315
2007-09-17 22:09:42,986 INFO objects found in image4: 324
2007-09-17 22:09:42,986 INFO starting objects parameters detection
2007-09-17 22:09:46,031 INFO starting paths detection
2007-09-17 22:09:47,716 INFO loading image 20060415_0954_c2.gif
```

```
2007-09-17 22:09:48,072 INFO loading image 20060415_1034_c2.gif
2007-09-17 22:09:48,421 INFO loading image 20060415_1106_c2.gif
2007-09-17 22:09:48,768 INFO loading image 20060415_1154_c2.gif
2007-09-17 22:09:49,121 INFO starting images subtractions
2007-09-17 22:09:49,550 INFO starting image blurring with sigma=0.7
2007-09-17 22:09:50,695 INFO starting objects detection
2007-09-17 22:09:54,979 INFO objects found in image1: 256
2007-09-17 22:09:54,979 INFO objects found in image2: 315
2007-09-17 22:09:54,980 INFO objects found in image3: 324
2007-09-17 22:09:54,980 INFO objects found in image4: 293
2007-09-17 22:09:54,980 INFO starting objects parameters detection
2007-09-17 22:09:58,043 INFO starting paths detection
2007-09-17 22:09:58,256 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-17 22:09:58,257 INFO [115 107  96  87  50  77  98 124]
--
2007-09-18 00:05:08,825 INFO starting objects detection
2007-09-18 00:05:13,171 INFO objects found in image1: 315
2007-09-18 00:05:13,171 INFO objects found in image2: 330
2007-09-18 00:05:13,171 INFO objects found in image3: 276
2007-09-18 00:05:13,172 INFO objects found in image4: 313
2007-09-18 00:05:13,172 INFO starting objects parameters detection
2007-09-18 00:05:16,143 INFO starting paths detection
2007-09-18 00:05:17,723 INFO loading image 20060509_0254_c2.gif
2007-09-18 00:05:18,097 INFO loading image 20060509_0306_c2.gif
2007-09-18 00:05:18,443 INFO loading image 20060509_0330_c2.gif
2007-09-18 00:05:18,806 INFO loading image 20060509_0354_c2.gif
2007-09-18 00:05:19,166 INFO starting images subtractions
2007-09-18 00:05:19,578 INFO starting image blurring with sigma=0.7
2007-09-18 00:05:20,720 INFO starting objects detection
2007-09-18 00:05:25,019 INFO objects found in image1: 343
2007-09-18 00:05:25,020 INFO objects found in image2: 276
2007-09-18 00:05:25,020 INFO objects found in image3: 313
2007-09-18 00:05:25,020 INFO objects found in image4: 351
2007-09-18 00:05:25,020 INFO starting objects parameters detection
2007-09-18 00:05:28,232 INFO starting paths detection
2007-09-18 00:05:28,456 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:05:28,457 INFO [463 468 480 492 153 163 182 202]
2007-09-18 00:05:37,961 INFO loading image 20060509_0306_c2.gif
2007-09-18 00:05:38,309 INFO loading image 20060509_0330_c2.gif
2007-09-18 00:05:38,660 INFO loading image 20060509_0354_c2.gif
2007-09-18 00:05:39,009 INFO loading image 20060509_0406_c2.gif
2007-09-18 00:05:39,360 INFO starting images subtractions
2007-09-18 00:05:39,761 INFO starting image blurring with sigma=0.7
2007-09-18 00:05:40,941 INFO starting objects detection
```

```
2007-09-18 00:05:45,127 INFO objects found in image1: 301
2007-09-18 00:05:45,128 INFO objects found in image2: 313
2007-09-18 00:05:45,128 INFO objects found in image3: 351
2007-09-18 00:05:45,128 INFO objects found in image4: 315
2007-09-18 00:05:45,128 INFO starting objects parameters detection
2007-09-18 00:05:48,376 INFO starting paths detection
2007-09-18 00:05:48,613 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:05:48,615 INFO [468 480 492 497 163 182 202 212]
--
2007-09-18 00:11:30,278 INFO starting objects detection
2007-09-18 00:11:34,574 INFO objects found in image1: 348
2007-09-18 00:11:34,575 INFO objects found in image2: 285
2007-09-18 00:11:34,575 INFO objects found in image3: 280
2007-09-18 00:11:34,575 INFO objects found in image4: 303
2007-09-18 00:11:34,575 INFO starting objects parameters detection
2007-09-18 00:11:37,484 INFO starting paths detection
2007-09-18 00:11:41,505 INFO loading image 20060510_1454_c2.gif
2007-09-18 00:11:41,854 INFO loading image 20060510_1506_c2.gif
2007-09-18 00:11:42,202 INFO loading image 20060510_1530_c2.gif
2007-09-18 00:11:42,555 INFO loading image 20060510_1554_c2.gif
2007-09-18 00:11:42,963 INFO starting images subtractions
2007-09-18 00:11:43,371 INFO starting image blurring with sigma=0.7
2007-09-18 00:11:44,520 INFO starting objects detection
2007-09-18 00:11:48,732 INFO objects found in image1: 284
2007-09-18 00:11:48,733 INFO objects found in image2: 280
2007-09-18 00:11:48,733 INFO objects found in image3: 303
2007-09-18 00:11:48,733 INFO objects found in image4: 299
2007-09-18 00:11:48,733 INFO starting objects parameters detection
2007-09-18 00:11:51,516 INFO starting paths detection
2007-09-18 00:11:51,713 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:11:51,714 INFO [436 441 452 463  81  90 108 126]
2007-09-18 00:11:59,210 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:11:59,211 INFO [436 441 452 459  81  90 108 119]
--
2007-09-18 00:17:57,598 INFO starting objects detection
2007-09-18 00:18:01,914 INFO objects found in image1: 338
2007-09-18 00:18:01,915 INFO objects found in image2: 314
2007-09-18 00:18:01,915 INFO objects found in image3: 283
2007-09-18 00:18:01,915 INFO objects found in image4: 318
2007-09-18 00:18:01,915 INFO starting objects parameters detection
2007-09-18 00:18:05,103 INFO starting paths detection
2007-09-18 00:18:07,204 INFO loading image 20060511_0454_c2.gif
2007-09-18 00:18:07,557 INFO loading image 20060511_0506_c2.gif
2007-09-18 00:18:07,976 INFO loading image 20060511_0530_c2.gif
```

```
2007-09-18 00:18:08,324 INFO loading image 20060511_0554_c2.gif
2007-09-18 00:18:08,677 INFO starting images subtractions
2007-09-18 00:18:09,120 INFO starting image blurring with sigma=0.7
2007-09-18 00:18:10,267 INFO starting objects detection
2007-09-18 00:18:14,564 INFO objects found in image1: 320
2007-09-18 00:18:14,565 INFO objects found in image2: 283
2007-09-18 00:18:14,565 INFO objects found in image3: 318
2007-09-18 00:18:14,565 INFO objects found in image4: 316
2007-09-18 00:18:14,565 INFO starting objects parameters detection
2007-09-18 00:18:17,489 INFO starting paths detection
2007-09-18 00:18:17,717 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:18:17,719 INFO [469 473 483 491  12  20  35  51]
2007-09-18 00:18:26,654 INFO loading image 20060511_0506_c2.gif
2007-09-18 00:18:27,000 INFO loading image 20060511_0530_c2.gif
2007-09-18 00:18:27,342 INFO loading image 20060511_0554_c2.gif
2007-09-18 00:18:27,693 INFO loading image 20060511_0606_c2.gif
2007-09-18 00:18:28,100 INFO starting images subtractions
2007-09-18 00:18:28,510 INFO starting image blurring with sigma=0.7
2007-09-18 00:18:29,656 INFO starting objects detection
2007-09-18 00:18:33,995 INFO objects found in image1: 288
2007-09-18 00:18:33,996 INFO objects found in image2: 318
2007-09-18 00:18:33,996 INFO objects found in image3: 316
2007-09-18 00:18:33,996 INFO objects found in image4: 323
2007-09-18 00:18:33,996 INFO starting objects parameters detection
2007-09-18 00:18:36,969 INFO starting paths detection
2007-09-18 00:18:37,195 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 00:18:37,196 INFO [473 483 491 496  20  35  51  58]
--
2007-09-18 01:11:30,913 INFO starting objects detection
2007-09-18 01:11:35,268 INFO objects found in image1: 328
2007-09-18 01:11:35,268 INFO objects found in image2: 294
2007-09-18 01:11:35,268 INFO objects found in image3: 260
2007-09-18 01:11:35,269 INFO objects found in image4: 417
2007-09-18 01:11:35,269 INFO starting objects parameters detection
2007-09-18 01:11:38,454 INFO starting paths detection
2007-09-18 01:11:40,122 INFO loading image 20060523_0454_c2.gif
2007-09-18 01:11:40,467 INFO loading image 20060523_0506_c2.gif
2007-09-18 01:11:40,815 INFO loading image 20060523_0530_c2.gif
2007-09-18 01:11:41,162 INFO loading image 20060523_0554_c2.gif
2007-09-18 01:11:41,506 INFO starting images subtractions
2007-09-18 01:11:41,925 INFO starting image blurring with sigma=0.7
2007-09-18 01:11:43,073 INFO starting objects detection
2007-09-18 01:11:47,338 INFO objects found in image1: 342
2007-09-18 01:11:47,338 INFO objects found in image2: 260
```

```
2007-09-18 01:11:47,338 INFO objects found in image3: 417
2007-09-18 01:11:47,338 INFO objects found in image4: 294
2007-09-18 01:11:47,339 INFO starting objects parameters detection
2007-09-18 01:11:50,665 INFO starting paths detection
2007-09-18 01:11:50,914 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:11:50,915 INFO [599 602 608 613  54  62  80  97]
2007-09-18 01:11:58,915 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:11:58,916 INFO [599 602 608 612  54  62  80  94]
2007-09-18 01:12:06,933 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:06,935 INFO [598 602 608 613  51  62  80  97]
2007-09-18 01:12:14,875 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:14,877 INFO [598 602 608 612  51  62  80  94]
2007-09-18 01:12:22,662 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:22,664 INFO [598 601 608 613  51  60  80  97]
2007-09-18 01:12:30,653 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:30,654 INFO [598 601 608 612  51  60  80  94]
2007-09-18 01:12:38,528 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:38,530 INFO [598 601 607 612  51  60  77  94]
2007-09-18 01:12:48,225 INFO loading image 20060523_0506_c2.gif
2007-09-18 01:12:48,591 INFO loading image 20060523_0530_c2.gif
2007-09-18 01:12:48,968 INFO loading image 20060523_0554_c2.gif
2007-09-18 01:12:49,333 INFO loading image 20060523_0606_c2.gif
2007-09-18 01:12:49,682 INFO starting images subtractions
2007-09-18 01:12:50,082 INFO starting image blurring with sigma=0.7
2007-09-18 01:12:51,231 INFO starting objects detection
2007-09-18 01:12:55,690 INFO objects found in image1: 315
2007-09-18 01:12:55,690 INFO objects found in image2: 417
2007-09-18 01:12:55,691 INFO objects found in image3: 294
2007-09-18 01:12:55,691 INFO objects found in image4: 312
2007-09-18 01:12:55,691 INFO starting objects parameters detection
2007-09-18 01:12:59,012 INFO starting paths detection
2007-09-18 01:12:59,282 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:12:59,283 INFO [602 608 613 616  63  80  97 106]
2007-09-18 01:13:07,300 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:13:07,301 INFO [602 608 613 616  63  80  97 103]
2007-09-18 01:13:15,161 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:13:15,162 INFO [601 608 613 616  60  80  97 106]
2007-09-18 01:13:23,098 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 01:13:23,099 INFO [601 608 613 616  60  80  97 103]
--
2007-09-18 02:07:00,701 INFO starting objects detection
2007-09-18 02:07:05,220 INFO objects found in image1: 330
2007-09-18 02:07:05,220 INFO objects found in image2: 365
2007-09-18 02:07:05,220 INFO objects found in image3: 295
```

```
2007-09-18 02:07:05,220 INFO objects found in image4: 295
2007-09-18 02:07:05,221 INFO starting objects parameters detection
2007-09-18 02:07:08,444 INFO starting paths detection
2007-09-18 02:07:10,073 INFO loading image 20060529_0712_c2.gif
2007-09-18 02:07:10,427 INFO loading image 20060529_0724_c2.gif
2007-09-18 02:07:10,766 INFO loading image 20060529_0736_c2.gif
2007-09-18 02:07:11,110 INFO loading image 20060529_0812_c2.gif
2007-09-18 02:07:11,452 INFO starting images subtractions
2007-09-18 02:07:11,892 INFO starting image blurring with sigma=0.7
2007-09-18 02:07:13,108 INFO starting objects detection
2007-09-18 02:07:17,305 INFO objects found in image1: 370
2007-09-18 02:07:17,305 INFO objects found in image2: 295
2007-09-18 02:07:17,306 INFO objects found in image3: 295
2007-09-18 02:07:17,306 INFO objects found in image4: 293
2007-09-18 02:07:17,306 INFO starting objects parameters detection
2007-09-18 02:07:20,461 INFO starting paths detection
2007-09-18 02:07:20,677 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:07:20,678 INFO [ 62  60  60  64  90 111 127 147]
--
2007-09-18 02:29:51,091 INFO starting objects detection
2007-09-18 02:29:55,341 INFO objects found in image1: 339
2007-09-18 02:29:55,341 INFO objects found in image2: 298
2007-09-18 02:29:55,342 INFO objects found in image3: 303
2007-09-18 02:29:55,342 INFO objects found in image4: 326
2007-09-18 02:29:55,342 INFO starting objects parameters detection
2007-09-18 02:29:58,472 INFO starting paths detection
2007-09-18 02:30:00,050 INFO loading image 20060530_2236_c2.gif
2007-09-18 02:30:00,402 INFO loading image 20060530_2312_c2.gif
2007-09-18 02:30:00,745 INFO loading image 20060530_2324_c2.gif
2007-09-18 02:30:01,089 INFO loading image 20060530_2336_c2.gif
2007-09-18 02:30:01,434 INFO starting images subtractions
2007-09-18 02:30:01,838 INFO starting image blurring with sigma=0.7
2007-09-18 02:30:03,014 INFO starting objects detection
2007-09-18 02:30:07,436 INFO objects found in image1: 311
2007-09-18 02:30:07,436 INFO objects found in image2: 303
2007-09-18 02:30:07,437 INFO objects found in image3: 326
2007-09-18 02:30:07,437 INFO objects found in image4: 350
2007-09-18 02:30:07,437 INFO starting objects parameters detection
2007-09-18 02:30:10,742 INFO starting paths detection
2007-09-18 02:30:10,982 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:30:10,983 INFO [662 667 669 670  10  32  40  48]
2007-09-18 02:30:20,230 INFO loading image 20060530_2312_c2.gif
2007-09-18 02:30:20,577 INFO loading image 20060530_2324_c2.gif
2007-09-18 02:30:20,924 INFO loading image 20060530_2336_c2.gif
```

```
2007-09-18 02:30:21,278 INFO loading image 20060531_0000_c2.gif
2007-09-18 02:30:21,622 INFO starting images subtractions
2007-09-18 02:30:22,029 INFO starting image blurring with sigma=0.7
2007-09-18 02:30:23,180 INFO starting objects detection
2007-09-18 02:30:27,410 INFO objects found in image1: 305
2007-09-18 02:30:27,410 INFO objects found in image2: 326
2007-09-18 02:30:27,410 INFO objects found in image3: 350
2007-09-18 02:30:27,411 INFO objects found in image4: 319
2007-09-18 02:30:27,411 INFO starting objects parameters detection
2007-09-18 02:30:30,623 INFO starting paths detection
2007-09-18 02:30:30,887 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:30:30,888 INFO [667 669 670 674  32  40  48  63]
2007-09-18 02:30:39,892 INFO loading image 20060530_2324_c2.gif
2007-09-18 02:30:40,242 INFO loading image 20060530_2336_c2.gif
2007-09-18 02:30:40,591 INFO loading image 20060531_0000_c2.gif
2007-09-18 02:30:40,937 INFO loading image 20060531_0012_c2.gif
2007-09-18 02:30:41,283 INFO starting images subtractions
2007-09-18 02:30:41,707 INFO starting image blurring with sigma=0.7
2007-09-18 02:30:42,861 INFO starting objects detection
2007-09-18 02:30:47,194 INFO objects found in image1: 318
2007-09-18 02:30:47,194 INFO objects found in image2: 350
2007-09-18 02:30:47,194 INFO objects found in image3: 319
2007-09-18 02:30:47,194 INFO objects found in image4: 291
2007-09-18 02:30:47,195 INFO starting objects parameters detection
2007-09-18 02:30:50,396 INFO starting paths detection
2007-09-18 02:30:50,653 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:30:50,654 INFO [668 670 674 675  40  48  63  71]
2007-09-18 02:31:00,041 INFO loading image 20060530_2336_c2.gif
2007-09-18 02:31:00,392 INFO loading image 20060531_0000_c2.gif
2007-09-18 02:31:00,762 INFO loading image 20060531_0012_c2.gif
2007-09-18 02:31:01,107 INFO loading image 20060531_0036_c2.gif
2007-09-18 02:31:01,461 INFO starting images subtractions
2007-09-18 02:31:01,897 INFO starting image blurring with sigma=0.7
2007-09-18 02:31:03,136 INFO starting objects detection
2007-09-18 02:31:07,839 INFO objects found in image1: 369
2007-09-18 02:31:07,839 INFO objects found in image2: 319
2007-09-18 02:31:07,839 INFO objects found in image3: 291
2007-09-18 02:31:07,840 INFO objects found in image4: 277
2007-09-18 02:31:07,840 INFO starting objects parameters detection
2007-09-18 02:31:11,025 INFO starting paths detection
2007-09-18 02:31:11,259 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:31:11,260 INFO [670 674 675 679  48  63  71  87]
2007-09-18 02:31:20,555 INFO loading image 20060531_0000_c2.gif
2007-09-18 02:31:20,911 INFO loading image 20060531_0012_c2.gif
```

```
2007-09-18 02:31:21,267 INFO loading image 20060531_0036_c2.gif
2007-09-18 02:31:21,625 INFO loading image 20060531_0048_c2.gif
2007-09-18 02:31:21,984 INFO starting images subtractions
2007-09-18 02:31:22,397 INFO starting image blurring with sigma=0.7
2007-09-18 02:31:23,570 INFO starting objects detection
2007-09-18 02:31:27,795 INFO objects found in image1: 322
2007-09-18 02:31:27,795 INFO objects found in image2: 291
2007-09-18 02:31:27,796 INFO objects found in image3: 277
2007-09-18 02:31:27,796 INFO objects found in image4: 349
2007-09-18 02:31:27,796 INFO starting objects parameters detection
2007-09-18 02:31:30,885 INFO starting paths detection
2007-09-18 02:31:31,094 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:31:31,095 INFO [674 675 679 680  63  71  87  95]
2007-09-18 02:31:40,491 INFO loading image 20060531_0012_c2.gif
2007-09-18 02:31:40,838 INFO loading image 20060531_0036_c2.gif
2007-09-18 02:31:41,184 INFO loading image 20060531_0048_c2.gif
2007-09-18 02:31:41,535 INFO loading image 20060531_0112_c2.gif
2007-09-18 02:31:41,883 INFO starting images subtractions
2007-09-18 02:31:42,301 INFO starting image blurring with sigma=0.7
2007-09-18 02:31:43,461 INFO starting objects detection
2007-09-18 02:31:47,649 INFO objects found in image1: 304
2007-09-18 02:31:47,650 INFO objects found in image2: 277
2007-09-18 02:31:47,650 INFO objects found in image3: 349
2007-09-18 02:31:47,650 INFO objects found in image4: 307
2007-09-18 02:31:47,650 INFO starting objects parameters detection
2007-09-18 02:31:50,731 INFO starting paths detection
2007-09-18 02:31:50,953 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:31:50,954 INFO [675 679 680 683  71  87  95 112]
2007-09-18 02:32:00,233 INFO loading image 20060531_0036_c2.gif
2007-09-18 02:32:00,576 INFO loading image 20060531_0048_c2.gif
2007-09-18 02:32:00,920 INFO loading image 20060531_0112_c2.gif
2007-09-18 02:32:01,263 INFO loading image 20060531_0124_c2.gif
2007-09-18 02:32:01,613 INFO starting images subtractions
2007-09-18 02:32:02,025 INFO starting image blurring with sigma=0.7
2007-09-18 02:32:03,207 INFO starting objects detection
2007-09-18 02:32:07,564 INFO objects found in image1: 260
2007-09-18 02:32:07,564 INFO objects found in image2: 349
2007-09-18 02:32:07,565 INFO objects found in image3: 307
2007-09-18 02:32:07,565 INFO objects found in image4: 349
2007-09-18 02:32:07,565 INFO starting objects parameters detection
2007-09-18 02:32:10,698 INFO starting paths detection
2007-09-18 02:32:10,934 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:32:10,935 INFO [679 680 683 685  87  95 112 120]
2007-09-18 02:32:20,266 INFO loading image 20060531_0048_c2.gif
```

```
2007-09-18 02:32:20,613 INFO loading image 20060531_0112_c2.gif
2007-09-18 02:32:20,965 INFO loading image 20060531_0124_c2.gif
2007-09-18 02:32:21,328 INFO loading image 20060531_0136_c2.gif
2007-09-18 02:32:21,675 INFO starting images subtractions
2007-09-18 02:32:22,137 INFO starting image blurring with sigma=0.7
2007-09-18 02:32:23,308 INFO starting objects detection
2007-09-18 02:32:27,625 INFO objects found in image1: 346
2007-09-18 02:32:27,625 INFO objects found in image2: 307
2007-09-18 02:32:27,626 INFO objects found in image3: 349
2007-09-18 02:32:27,626 INFO objects found in image4: 306
2007-09-18 02:32:27,626 INFO starting objects parameters detection
2007-09-18 02:32:30,881 INFO starting paths detection
2007-09-18 02:32:31,129 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:32:31,130 INFO [680 683 685 686  95 112 120 128]
2007-09-18 02:32:40,754 INFO loading image 20060531_0112_c2.gif
2007-09-18 02:32:41,101 INFO loading image 20060531_0124_c2.gif
2007-09-18 02:32:41,445 INFO loading image 20060531_0136_c2.gif
2007-09-18 02:32:41,785 INFO loading image 20060531_0200_c2.gif
2007-09-18 02:32:42,129 INFO starting images subtractions
2007-09-18 02:32:42,554 INFO starting image blurring with sigma=0.7
2007-09-18 02:32:43,716 INFO starting objects detection
2007-09-18 02:32:47,958 INFO objects found in image1: 302
2007-09-18 02:32:47,959 INFO objects found in image2: 349
2007-09-18 02:32:47,959 INFO objects found in image3: 306
2007-09-18 02:32:47,959 INFO objects found in image4: 353
2007-09-18 02:32:47,959 INFO starting objects parameters detection
2007-09-18 02:32:51,179 INFO starting paths detection
2007-09-18 02:32:51,433 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:32:51,434 INFO [683 685 686 689 112 120 128 145]
2007-09-18 02:33:00,618 INFO loading image 20060531_0124_c2.gif
2007-09-18 02:33:00,965 INFO loading image 20060531_0136_c2.gif
2007-09-18 02:33:01,315 INFO loading image 20060531_0200_c2.gif
2007-09-18 02:33:01,663 INFO loading image 20060531_0212_c2.gif
2007-09-18 02:33:02,014 INFO starting images subtractions
2007-09-18 02:33:02,442 INFO starting image blurring with sigma=0.7
2007-09-18 02:33:03,657 INFO starting objects detection
2007-09-18 02:33:07,966 INFO objects found in image1: 362
2007-09-18 02:33:07,966 INFO objects found in image2: 306
2007-09-18 02:33:07,966 INFO objects found in image3: 353
2007-09-18 02:33:07,966 INFO objects found in image4: 310
2007-09-18 02:33:07,967 INFO starting objects parameters detection
2007-09-18 02:33:11,263 INFO starting paths detection
2007-09-18 02:33:11,523 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:33:11,524 INFO [685 686 689 691 120 128 145 154]
```

```
2007-09-18 02:33:20,863 INFO loading image 20060531_0136_c2.gif
2007-09-18 02:33:21,324 INFO loading image 20060531_0200_c2.gif
2007-09-18 02:33:21,670 INFO loading image 20060531_0212_c2.gif
2007-09-18 02:33:22,018 INFO loading image 20060531_0236_c2.gif
2007-09-18 02:33:22,369 INFO starting images subtractions
2007-09-18 02:33:22,776 INFO starting image blurring with sigma=0.7
2007-09-18 02:33:23,934 INFO starting objects detection
2007-09-18 02:33:28,176 INFO objects found in image1: 295
2007-09-18 02:33:28,176 INFO objects found in image2: 353
2007-09-18 02:33:28,177 INFO objects found in image3: 310
2007-09-18 02:33:28,177 INFO objects found in image4: 302
2007-09-18 02:33:28,177 INFO starting objects parameters detection
2007-09-18 02:33:31,304 INFO starting paths detection
2007-09-18 02:33:31,551 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:33:31,552 INFO [686 689 691 693 128 145 154 172]
2007-09-18 02:33:41,089 INFO loading image 20060531_0200_c2.gif
2007-09-18 02:33:41,434 INFO loading image 20060531_0212_c2.gif
2007-09-18 02:33:41,779 INFO loading image 20060531_0236_c2.gif
2007-09-18 02:33:42,122 INFO loading image 20060531_0248_c2.gif
2007-09-18 02:33:42,466 INFO starting images subtractions
2007-09-18 02:33:42,869 INFO starting image blurring with sigma=0.7
2007-09-18 02:33:44,019 INFO starting objects detection
2007-09-18 02:33:48,307 INFO objects found in image1: 343
2007-09-18 02:33:48,307 INFO objects found in image2: 310
2007-09-18 02:33:48,307 INFO objects found in image3: 302
2007-09-18 02:33:48,308 INFO objects found in image4: 306
2007-09-18 02:33:48,308 INFO starting objects parameters detection
2007-09-18 02:33:51,423 INFO starting paths detection
2007-09-18 02:33:51,646 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:33:51,647 INFO [689 691 693 695 145 154 172 181]
2007-09-18 02:34:01,202 INFO loading image 20060531_0212_c2.gif
2007-09-18 02:34:01,545 INFO loading image 20060531_0236_c2.gif
2007-09-18 02:34:01,890 INFO loading image 20060531_0248_c2.gif
2007-09-18 02:34:02,231 INFO loading image 20060531_0312_c2.gif
2007-09-18 02:34:02,579 INFO starting images subtractions
2007-09-18 02:34:02,980 INFO starting image blurring with sigma=0.7
2007-09-18 02:34:04,146 INFO starting objects detection
2007-09-18 02:34:08,607 INFO objects found in image1: 305
2007-09-18 02:34:08,607 INFO objects found in image2: 302
2007-09-18 02:34:08,607 INFO objects found in image3: 306
2007-09-18 02:34:08,608 INFO objects found in image4: 301
2007-09-18 02:34:08,608 INFO starting objects parameters detection
2007-09-18 02:34:11,586 INFO starting paths detection
2007-09-18 02:34:11,802 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
```

```
2007-09-18 02:34:11,803 INFO [691 693 695 697 154 172 181 199]
2007-09-18 02:34:21,359 INFO loading image 20060531_0236_c2.gif
2007-09-18 02:34:21,711 INFO loading image 20060531_0248_c2.gif
2007-09-18 02:34:22,058 INFO loading image 20060531_0312_c2.gif
2007-09-18 02:34:22,406 INFO loading image 20060531_0324_c2.gif
2007-09-18 02:34:22,757 INFO starting images subtractions
2007-09-18 02:34:23,152 INFO starting image blurring with sigma=0.7
2007-09-18 02:34:24,317 INFO starting objects detection
2007-09-18 02:34:28,539 INFO objects found in image1: 291
2007-09-18 02:34:28,539 INFO objects found in image2: 306
2007-09-18 02:34:28,540 INFO objects found in image3: 301
2007-09-18 02:34:28,540 INFO objects found in image4: 319
2007-09-18 02:34:28,540 INFO starting objects parameters detection
2007-09-18 02:34:31,531 INFO starting paths detection
2007-09-18 02:34:31,745 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:34:31,746 INFO [693 695 697 698 172 181 199 208]
2007-09-18 02:34:41,218 INFO loading image 20060531_0248_c2.gif
2007-09-18 02:34:41,571 INFO loading image 20060531_0312_c2.gif
2007-09-18 02:34:41,915 INFO loading image 20060531_0324_c2.gif
2007-09-18 02:34:42,259 INFO loading image 20060531_0336_c2.gif
2007-09-18 02:34:42,610 INFO starting images subtractions
2007-09-18 02:34:43,033 INFO starting image blurring with sigma=0.7
2007-09-18 02:34:44,168 INFO starting objects detection
2007-09-18 02:34:48,362 INFO objects found in image1: 303
2007-09-18 02:34:48,363 INFO objects found in image2: 301
2007-09-18 02:34:48,363 INFO objects found in image3: 319
2007-09-18 02:34:48,363 INFO objects found in image4: 313
2007-09-18 02:34:48,363 INFO starting objects parameters detection
2007-09-18 02:34:51,422 INFO starting paths detection
2007-09-18 02:34:51,645 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:34:51,646 INFO [694 697 698 699 181 199 208 217]
2007-09-18 02:35:01,042 INFO loading image 20060531_0312_c2.gif
2007-09-18 02:35:01,389 INFO loading image 20060531_0324_c2.gif
2007-09-18 02:35:01,732 INFO loading image 20060531_0336_c2.gif
2007-09-18 02:35:02,093 INFO loading image 20060531_0400_c2.gif
2007-09-18 02:35:02,444 INFO starting images subtractions
2007-09-18 02:35:02,885 INFO starting image blurring with sigma=0.7
2007-09-18 02:35:04,040 INFO starting objects detection
2007-09-18 02:35:08,484 INFO objects found in image1: 302
2007-09-18 02:35:08,485 INFO objects found in image2: 319
2007-09-18 02:35:08,485 INFO objects found in image3: 313
2007-09-18 02:35:08,485 INFO objects found in image4: 307
2007-09-18 02:35:08,485 INFO starting objects parameters detection
2007-09-18 02:35:11,556 INFO starting paths detection
```

```
2007-09-18 02:35:11,788 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:35:11,789 INFO [697 698 699 701 199 208 217 237]
2007-09-18 02:35:21,518 INFO loading image 20060531_0324_c2.gif
2007-09-18 02:35:21,865 INFO loading image 20060531_0336_c2.gif
2007-09-18 02:35:22,212 INFO loading image 20060531_0400_c2.gif
2007-09-18 02:35:22,559 INFO loading image 20060531_0412_c2.gif
2007-09-18 02:35:22,903 INFO starting images subtractions
2007-09-18 02:35:23,344 INFO starting image blurring with sigma=0.7
2007-09-18 02:35:24,556 INFO starting objects detection
2007-09-18 02:35:28,737 INFO objects found in image1: 305
2007-09-18 02:35:28,738 INFO objects found in image2: 313
2007-09-18 02:35:28,738 INFO objects found in image3: 307
2007-09-18 02:35:28,738 INFO objects found in image4: 320
2007-09-18 02:35:28,739 INFO starting objects parameters detection
2007-09-18 02:35:31,790 INFO starting paths detection
2007-09-18 02:35:32,025 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:35:32,026 INFO [698 699 701 702 208 217 237 246]
--
2007-09-18 02:59:12,020 INFO starting objects detection
2007-09-18 02:59:16,294 INFO objects found in image1: 337
2007-09-18 02:59:16,294 INFO objects found in image2: 303
2007-09-18 02:59:16,294 INFO objects found in image3: 311
2007-09-18 02:59:16,295 INFO objects found in image4: 315
2007-09-18 02:59:16,295 INFO starting objects parameters detection
2007-09-18 02:59:19,461 INFO starting paths detection
2007-09-18 02:59:21,322 INFO loading image 20060601_2136_c2.gif
2007-09-18 02:59:21,665 INFO loading image 20060601_2148_c2.gif
2007-09-18 02:59:22,015 INFO loading image 20060601_2200_c2.gif
2007-09-18 02:59:22,365 INFO loading image 20060601_2212_c2.gif
2007-09-18 02:59:22,711 INFO starting images subtractions
2007-09-18 02:59:23,132 INFO starting image blurring with sigma=0.7
2007-09-18 02:59:24,356 INFO starting objects detection
2007-09-18 02:59:28,660 INFO objects found in image1: 291
2007-09-18 02:59:28,660 INFO objects found in image2: 311
2007-09-18 02:59:28,660 INFO objects found in image3: 315
2007-09-18 02:59:28,660 INFO objects found in image4: 295
2007-09-18 02:59:28,661 INFO starting objects parameters detection
2007-09-18 02:59:31,851 INFO starting paths detection
2007-09-18 02:59:32,070 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:59:32,071 INFO [673 675 676 677   7  15  22  31]
2007-09-18 02:59:42,067 INFO loading image 20060601_2148_c2.gif
2007-09-18 02:59:42,421 INFO loading image 20060601_2200_c2.gif
2007-09-18 02:59:42,767 INFO loading image 20060601_2212_c2.gif
2007-09-18 02:59:43,118 INFO loading image 20060601_2224_c2.gif
```

```
2007-09-18 02:59:43,465 INFO starting images subtractions
2007-09-18 02:59:43,907 INFO starting image blurring with sigma=0.7
2007-09-18 02:59:45,170 INFO starting objects detection
2007-09-18 02:59:49,414 INFO objects found in image1: 318
2007-09-18 02:59:49,415 INFO objects found in image2: 315
2007-09-18 02:59:49,415 INFO objects found in image3: 295
2007-09-18 02:59:49,415 INFO objects found in image4: 325
2007-09-18 02:59:49,415 INFO starting objects parameters detection
2007-09-18 02:59:52,580 INFO starting paths detection
2007-09-18 02:59:52,799 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 02:59:52,800 INFO [675 676 677 679  15  22  31  39]
--
2007-09-18 12:15:36,368 INFO starting objects detection
2007-09-18 12:15:40,600 INFO objects found in image1: 345
2007-09-18 12:15:40,601 INFO objects found in image2: 299
2007-09-18 12:15:40,601 INFO objects found in image3: 333
2007-09-18 12:15:40,601 INFO objects found in image4: 303
2007-09-18 12:15:40,601 INFO starting objects parameters detection
2007-09-18 12:15:43,809 INFO starting paths detection
2007-09-18 12:15:45,378 INFO loading image 20060902_0200_c2.gif
2007-09-18 12:15:45,733 INFO loading image 20060902_0212_c2.gif
2007-09-18 12:15:46,080 INFO loading image 20060902_0224_c2.gif
2007-09-18 12:15:46,426 INFO loading image 20060902_0248_c2.gif
2007-09-18 12:15:46,772 INFO starting images subtractions
2007-09-18 12:15:47,179 INFO starting image blurring with sigma=0.7
2007-09-18 12:15:48,414 INFO starting objects detection
2007-09-18 12:15:52,613 INFO objects found in image1: 309
2007-09-18 12:15:52,614 INFO objects found in image2: 333
2007-09-18 12:15:52,614 INFO objects found in image3: 303
2007-09-18 12:15:52,614 INFO objects found in image4: 312
2007-09-18 12:15:52,614 INFO starting objects parameters detection
2007-09-18 12:15:55,795 INFO starting paths detection
2007-09-18 12:15:56,021 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 12:15:56,022 INFO [942 943 945 948 980 955 931 913]
--
2007-09-18 16:49:16,528 INFO starting objects detection
2007-09-18 16:49:20,656 INFO objects found in image1: 439
2007-09-18 16:49:20,657 INFO objects found in image2: 383
2007-09-18 16:49:20,657 INFO objects found in image3: 336
2007-09-18 16:49:20,657 INFO objects found in image4: 364
2007-09-18 16:49:20,657 INFO starting objects parameters detection
2007-09-18 16:49:24,475 INFO starting paths detection
2007-09-18 16:49:26,261 INFO loading image 20061106_0554_c2.gif
2007-09-18 16:49:26,643 INFO loading image 20061106_0606_c2.gif
```

```
2007-09-18 16:49:27,014 INFO loading image 20061106_0630_c2.gif
2007-09-18 16:49:27,383 INFO loading image 20061106_0654_c2.gif
2007-09-18 16:49:27,775 INFO starting images subtractions
2007-09-18 16:49:28,220 INFO starting image blurring with sigma=0.7
2007-09-18 16:49:29,367 INFO starting objects detection
2007-09-18 16:49:33,483 INFO objects found in image1: 378
2007-09-18 16:49:33,484 INFO objects found in image2: 336
2007-09-18 16:49:33,484 INFO objects found in image3: 364
2007-09-18 16:49:33,484 INFO objects found in image4: 422
2007-09-18 16:49:33,484 INFO starting objects parameters detection
2007-09-18 16:49:37,150 INFO starting paths detection
2007-09-18 16:49:37,482 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 16:49:37,483 INFO [ 51  51  50  48  43  63  89 107]
--
2007-09-18 17:24:11,979 INFO starting objects detection
2007-09-18 17:24:16,139 INFO objects found in image1: 347
2007-09-18 17:24:16,139 INFO objects found in image2: 328
2007-09-18 17:24:16,140 INFO objects found in image3: 366
2007-09-18 17:24:16,140 INFO objects found in image4: 330
2007-09-18 17:24:16,140 INFO starting objects parameters detection
2007-09-18 17:24:19,692 INFO starting paths detection
2007-09-18 17:24:21,312 INFO loading image 20061114_0306_c2.gif
2007-09-18 17:24:21,658 INFO loading image 20061114_0326_c2.gif
2007-09-18 17:24:22,006 INFO loading image 20061114_0350_c2.gif
2007-09-18 17:24:22,345 INFO loading image 20061114_0406_c2.gif
2007-09-18 17:24:22,685 INFO starting images subtractions
2007-09-18 17:24:23,099 INFO starting image blurring with sigma=0.7
2007-09-18 17:24:24,182 INFO starting objects detection
2007-09-18 17:24:28,299 INFO objects found in image1: 316
2007-09-18 17:24:28,300 INFO objects found in image2: 366
2007-09-18 17:24:28,300 INFO objects found in image3: 330
2007-09-18 17:24:28,301 INFO objects found in image4: 337
2007-09-18 17:24:28,301 INFO starting objects parameters detection
2007-09-18 17:24:31,812 INFO starting paths detection
2007-09-18 17:24:32,077 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 17:24:32,078 INFO [542 533 522 515  59  77  98 113]
--
2007-09-18 20:06:19,499 INFO starting objects detection
2007-09-18 20:06:23,616 INFO objects found in image1: 386
2007-09-18 20:06:23,616 INFO objects found in image2: 437
2007-09-18 20:06:23,617 INFO objects found in image3: 393
2007-09-18 20:06:23,617 INFO objects found in image4: 365
2007-09-18 20:06:23,617 INFO starting objects parameters detection
2007-09-18 20:06:27,703 INFO starting paths detection
```

```
2007-09-18 20:06:29,563 INFO loading image 20061212_0836_c2.gif
2007-09-18 20:06:29,912 INFO loading image 20061212_0900_c2.gif
2007-09-18 20:06:30,258 INFO loading image 20061212_0912_c2.gif
2007-09-18 20:06:30,603 INFO loading image 20061212_0936_c2.gif
2007-09-18 20:06:30,954 INFO starting images subtractions
2007-09-18 20:06:31,365 INFO starting image blurring with sigma=0.7
2007-09-18 20:06:32,690 INFO starting objects detection
2007-09-18 20:06:36,789 INFO objects found in image1: 420
2007-09-18 20:06:36,790 INFO objects found in image2: 393
2007-09-18 20:06:36,790 INFO objects found in image3: 365
2007-09-18 20:06:36,790 INFO objects found in image4: 431
2007-09-18 20:06:36,790 INFO starting objects parameters detection
2007-09-18 20:06:40,818 INFO starting paths detection
2007-09-18 20:06:41,283 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 20:06:41,284 INFO [188 190 191 194  43  63  73  93]
2007-09-18 20:06:51,526 INFO loading image 20061212_0900_c2.gif
2007-09-18 20:06:51,905 INFO loading image 20061212_0912_c2.gif
2007-09-18 20:06:52,255 INFO loading image 20061212_0936_c2.gif
2007-09-18 20:06:52,619 INFO loading image 20061212_0948_c2.gif
2007-09-18 20:06:52,966 INFO starting images subtractions
2007-09-18 20:06:53,361 INFO starting image blurring with sigma=0.7
2007-09-18 20:06:54,558 INFO starting objects detection
2007-09-18 20:06:58,595 INFO objects found in image1: 401
2007-09-18 20:06:58,596 INFO objects found in image2: 365
2007-09-18 20:06:58,596 INFO objects found in image3: 431
2007-09-18 20:06:58,596 INFO objects found in image4: 419
2007-09-18 20:06:58,596 INFO starting objects parameters detection
2007-09-18 20:07:02,761 INFO starting paths detection
2007-09-18 20:07:03,446 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-18 20:07:03,447 INFO [190 191 194 195  63  73  93 103]
```

B.2 S2

```
2007-09-12 02:29:05,416 INFO starting objects detection
2007-09-12 02:29:09,626 INFO objects found in image1: 339
2007-09-12 02:29:09,626 INFO objects found in image2: 298
2007-09-12 02:29:09,626 INFO objects found in image3: 303
2007-09-12 02:29:09,627 INFO objects found in image4: 326
2007-09-12 02:29:09,627 INFO starting objects parameters detection
2007-09-12 02:29:12,751 INFO starting paths detection
2007-09-12 02:29:14,334 INFO loading image 20060530_2236_c2.gif
```

```
2007-09-12 02:29:14,686 INFO loading image 20060530_2312_c2.gif
2007-09-12 02:29:15,051 INFO loading image 20060530_2324_c2.gif
2007-09-12 02:29:15,402 INFO loading image 20060530_2336_c2.gif
2007-09-12 02:29:15,756 INFO starting images subtractions
2007-09-12 02:29:16,187 INFO starting image blurring with sigma=0.7
2007-09-12 02:29:17,514 INFO starting objects detection
2007-09-12 02:29:21,633 INFO objects found in image1: 311
2007-09-12 02:29:21,633 INFO objects found in image2: 303
2007-09-12 02:29:21,633 INFO objects found in image3: 326
2007-09-12 02:29:21,634 INFO objects found in image4: 350
2007-09-12 02:29:21,634 INFO starting objects parameters detection
2007-09-12 02:29:24,814 INFO starting paths detection
2007-09-12 02:29:25,081 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:29:25,083 INFO [662 667 669 670  10  32  40  48]
2007-09-12 02:29:37,393 INFO loading image 20060530_2312_c2.gif
2007-09-12 02:29:37,741 INFO loading image 20060530_2324_c2.gif
2007-09-12 02:29:38,088 INFO loading image 20060530_2336_c2.gif
2007-09-12 02:29:38,440 INFO loading image 20060531_0000_c2.gif
2007-09-12 02:29:38,783 INFO starting images subtractions
2007-09-12 02:29:39,193 INFO starting image blurring with sigma=0.7
2007-09-12 02:29:40,389 INFO starting objects detection
2007-09-12 02:29:44,606 INFO objects found in image1: 305
2007-09-12 02:29:44,606 INFO objects found in image2: 326
2007-09-12 02:29:44,606 INFO objects found in image3: 350
2007-09-12 02:29:44,606 INFO objects found in image4: 319
2007-09-12 02:29:44,607 INFO starting objects parameters detection
2007-09-12 02:29:47,922 INFO starting paths detection
2007-09-12 02:29:48,182 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:29:48,183 INFO [667 669 670 674  32  40  48  63]
--
2007-09-12 02:30:00,182 INFO starting objects detection
2007-09-12 02:30:04,549 INFO objects found in image1: 318
2007-09-12 02:30:04,549 INFO objects found in image2: 350
2007-09-12 02:30:04,549 INFO objects found in image3: 319
2007-09-12 02:30:04,550 INFO objects found in image4: 291
2007-09-12 02:30:04,550 INFO starting objects parameters detection
2007-09-12 02:30:07,808 INFO starting paths detection
2007-09-12 02:30:09,393 INFO loading image 20060530_2336_c2.gif
2007-09-12 02:30:09,735 INFO loading image 20060531_0000_c2.gif
2007-09-12 02:30:10,082 INFO loading image 20060531_0012_c2.gif
2007-09-12 02:30:10,429 INFO loading image 20060531_0036_c2.gif
2007-09-12 02:30:10,769 INFO starting images subtractions
2007-09-12 02:30:11,187 INFO starting image blurring with sigma=0.7
2007-09-12 02:30:12,418 INFO starting objects detection
```

```
2007-09-12 02:30:16,561 INFO objects found in image1: 369
2007-09-12 02:30:16,562 INFO objects found in image2: 319
2007-09-12 02:30:16,562 INFO objects found in image3: 291
2007-09-12 02:30:16,562 INFO objects found in image4: 277
2007-09-12 02:30:16,562 INFO starting objects parameters detection
2007-09-12 02:30:19,776 INFO starting paths detection
2007-09-12 02:30:19,998 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:30:19,999 INFO [670 674 675 679  48  63  71  87]
2007-09-12 02:30:29,124 INFO loading image 20060531_0000_c2.gif
2007-09-12 02:30:29,481 INFO loading image 20060531_0012_c2.gif
2007-09-12 02:30:29,852 INFO loading image 20060531_0036_c2.gif
2007-09-12 02:30:30,206 INFO loading image 20060531_0048_c2.gif
2007-09-12 02:30:30,562 INFO starting images subtractions
2007-09-12 02:30:30,978 INFO starting image blurring with sigma=0.7
2007-09-12 02:30:32,118 INFO starting objects detection
2007-09-12 02:30:36,526 INFO objects found in image1: 322
2007-09-12 02:30:36,526 INFO objects found in image2: 291
2007-09-12 02:30:36,526 INFO objects found in image3: 277
2007-09-12 02:30:36,527 INFO objects found in image4: 349
2007-09-12 02:30:36,527 INFO starting objects parameters detection
2007-09-12 02:30:39,621 INFO starting paths detection
2007-09-12 02:30:39,825 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:30:39,826 INFO [674 675 679 680  63  71  87  95]
2007-09-12 02:30:49,053 INFO loading image 20060531_0012_c2.gif
2007-09-12 02:30:49,418 INFO loading image 20060531_0036_c2.gif
2007-09-12 02:30:49,775 INFO loading image 20060531_0048_c2.gif
2007-09-12 02:30:50,133 INFO loading image 20060531_0112_c2.gif
2007-09-12 02:30:50,497 INFO starting images subtractions
2007-09-12 02:30:50,932 INFO starting image blurring with sigma=0.7
2007-09-12 02:30:52,072 INFO starting objects detection
2007-09-12 02:30:56,267 INFO objects found in image1: 304
2007-09-12 02:30:56,268 INFO objects found in image2: 277
2007-09-12 02:30:56,268 INFO objects found in image3: 349
2007-09-12 02:30:56,268 INFO objects found in image4: 307
2007-09-12 02:30:56,268 INFO starting objects parameters detection
2007-09-12 02:30:59,362 INFO starting paths detection
2007-09-12 02:30:59,583 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:30:59,584 INFO [675 679 680 683  71  87  95 112]
2007-09-12 02:31:09,000 INFO loading image 20060531_0036_c2.gif
2007-09-12 02:31:09,349 INFO loading image 20060531_0048_c2.gif
2007-09-12 02:31:09,701 INFO loading image 20060531_0112_c2.gif
2007-09-12 02:31:10,045 INFO loading image 20060531_0124_c2.gif
2007-09-12 02:31:10,390 INFO starting images subtractions
2007-09-12 02:31:10,798 INFO starting image blurring with sigma=0.7
```

```
2007-09-12 02:31:11,944 INFO starting objects detection
2007-09-12 02:31:16,141 INFO objects found in image1: 260
2007-09-12 02:31:16,142 INFO objects found in image2: 349
2007-09-12 02:31:16,142 INFO objects found in image3: 307
2007-09-12 02:31:16,142 INFO objects found in image4: 349
2007-09-12 02:31:16,142 INFO starting objects parameters detection
2007-09-12 02:31:19,302 INFO starting paths detection
2007-09-12 02:31:19,535 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:31:19,536 INFO [679 680 683 685  87  95 112 120]
2007-09-12 02:31:28,767 INFO loading image 20060531_0048_c2.gif
2007-09-12 02:31:29,116 INFO loading image 20060531_0112_c2.gif
2007-09-12 02:31:29,469 INFO loading image 20060531_0124_c2.gif
2007-09-12 02:31:29,820 INFO loading image 20060531_0136_c2.gif
2007-09-12 02:31:30,164 INFO starting images subtractions
2007-09-12 02:31:30,564 INFO starting image blurring with sigma=0.7
2007-09-12 02:31:31,707 INFO starting objects detection
2007-09-12 02:31:35,954 INFO objects found in image1: 346
2007-09-12 02:31:35,955 INFO objects found in image2: 307
2007-09-12 02:31:35,955 INFO objects found in image3: 349
2007-09-12 02:31:35,955 INFO objects found in image4: 306
2007-09-12 02:31:35,955 INFO starting objects parameters detection
2007-09-12 02:31:39,226 INFO starting paths detection
2007-09-12 02:31:39,474 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:31:39,475 INFO [680 683 685 686  95 112 120 128]
2007-09-12 02:31:48,650 INFO loading image 20060531_0112_c2.gif
2007-09-12 02:31:49,004 INFO loading image 20060531_0124_c2.gif
2007-09-12 02:31:49,356 INFO loading image 20060531_0136_c2.gif
2007-09-12 02:31:49,708 INFO loading image 20060531_0200_c2.gif
2007-09-12 02:31:50,059 INFO starting images subtractions
2007-09-12 02:31:50,469 INFO starting image blurring with sigma=0.7
2007-09-12 02:31:51,610 INFO starting objects detection
2007-09-12 02:31:55,813 INFO objects found in image1: 302
2007-09-12 02:31:55,814 INFO objects found in image2: 349
2007-09-12 02:31:55,814 INFO objects found in image3: 306
2007-09-12 02:31:55,814 INFO objects found in image4: 353
2007-09-12 02:31:55,814 INFO starting objects parameters detection
2007-09-12 02:31:59,065 INFO starting paths detection
2007-09-12 02:31:59,311 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:31:59,312 INFO [683 685 686 689 112 120 128 145]
2007-09-12 02:32:08,606 INFO loading image 20060531_0124_c2.gif
2007-09-12 02:32:08,951 INFO loading image 20060531_0136_c2.gif
2007-09-12 02:32:09,312 INFO loading image 20060531_0200_c2.gif
2007-09-12 02:32:09,659 INFO loading image 20060531_0212_c2.gif
2007-09-12 02:32:10,009 INFO starting images subtractions
```

```
2007-09-12 02:32:10,448 INFO starting image blurring with sigma=0.7
2007-09-12 02:32:11,603 INFO starting objects detection
2007-09-12 02:32:15,817 INFO objects found in image1: 362
2007-09-12 02:32:15,818 INFO objects found in image2: 306
2007-09-12 02:32:15,818 INFO objects found in image3: 353
2007-09-12 02:32:15,818 INFO objects found in image4: 310
2007-09-12 02:32:15,819 INFO starting objects parameters detection
2007-09-12 02:32:19,115 INFO starting paths detection
2007-09-12 02:32:19,380 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:32:19,380 INFO [685 686 689 691 120 128 145 154]
2007-09-12 02:32:28,530 INFO loading image 20060531_0136_c2.gif
2007-09-12 02:32:28,912 INFO loading image 20060531_0200_c2.gif
2007-09-12 02:32:29,258 INFO loading image 20060531_0212_c2.gif
2007-09-12 02:32:29,608 INFO loading image 20060531_0236_c2.gif
2007-09-12 02:32:29,954 INFO starting images subtractions
2007-09-12 02:32:30,364 INFO starting image blurring with sigma=0.7
2007-09-12 02:32:31,507 INFO starting objects detection
2007-09-12 02:32:35,715 INFO objects found in image1: 295
2007-09-12 02:32:35,715 INFO objects found in image2: 353
2007-09-12 02:32:35,716 INFO objects found in image3: 310
2007-09-12 02:32:35,716 INFO objects found in image4: 302
2007-09-12 02:32:35,716 INFO starting objects parameters detection
2007-09-12 02:32:38,878 INFO starting paths detection
2007-09-12 02:32:39,112 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:32:39,113 INFO [686 689 691 693 128 145 154 172]
2007-09-12 02:32:48,380 INFO loading image 20060531_0200_c2.gif
2007-09-12 02:32:48,728 INFO loading image 20060531_0212_c2.gif
2007-09-12 02:32:49,095 INFO loading image 20060531_0236_c2.gif
2007-09-12 02:32:49,448 INFO loading image 20060531_0248_c2.gif
2007-09-12 02:32:49,795 INFO starting images subtractions
2007-09-12 02:32:50,206 INFO starting image blurring with sigma=0.7
2007-09-12 02:32:51,343 INFO starting objects detection
2007-09-12 02:32:55,600 INFO objects found in image1: 343
2007-09-12 02:32:55,601 INFO objects found in image2: 310
2007-09-12 02:32:55,601 INFO objects found in image3: 302
2007-09-12 02:32:55,601 INFO objects found in image4: 306
2007-09-12 02:32:55,601 INFO starting objects parameters detection
2007-09-12 02:32:59,058 INFO starting paths detection
2007-09-12 02:32:59,281 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:32:59,282 INFO [689 691 693 695 145 154 172 181]
2007-09-12 02:33:08,758 INFO loading image 20060531_0212_c2.gif
2007-09-12 02:33:09,149 INFO loading image 20060531_0236_c2.gif
2007-09-12 02:33:09,497 INFO loading image 20060531_0248_c2.gif
2007-09-12 02:33:09,839 INFO loading image 20060531_0312_c2.gif
```

```
2007-09-12 02:33:10,229 INFO starting images subtractions
2007-09-12 02:33:10,646 INFO starting image blurring with sigma=0.7
2007-09-12 02:33:11,855 INFO starting objects detection
2007-09-12 02:33:16,245 INFO objects found in image1: 305
2007-09-12 02:33:16,246 INFO objects found in image2: 302
2007-09-12 02:33:16,246 INFO objects found in image3: 306
2007-09-12 02:33:16,246 INFO objects found in image4: 301
2007-09-12 02:33:16,246 INFO starting objects parameters detection
2007-09-12 02:33:19,359 INFO starting paths detection
2007-09-12 02:33:19,571 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:33:19,572 INFO [691 693 695 697 154 172 181 199]
2007-09-12 02:33:29,004 INFO loading image 20060531_0236_c2.gif
2007-09-12 02:33:29,348 INFO loading image 20060531_0248_c2.gif
2007-09-12 02:33:29,695 INFO loading image 20060531_0312_c2.gif
2007-09-12 02:33:30,034 INFO loading image 20060531_0324_c2.gif
2007-09-12 02:33:30,380 INFO starting images subtractions
2007-09-12 02:33:30,790 INFO starting image blurring with sigma=0.7
2007-09-12 02:33:32,033 INFO starting objects detection
2007-09-12 02:33:36,665 INFO objects found in image1: 291
2007-09-12 02:33:36,665 INFO objects found in image2: 306
2007-09-12 02:33:36,665 INFO objects found in image3: 301
2007-09-12 02:33:36,666 INFO objects found in image4: 319
2007-09-12 02:33:36,666 INFO starting objects parameters detection
2007-09-12 02:33:39,784 INFO starting paths detection
2007-09-12 02:33:39,998 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:33:39,999 INFO [693 695 697 698 172 181 199 208]
2007-09-12 02:33:49,217 INFO loading image 20060531_0248_c2.gif
2007-09-12 02:33:49,569 INFO loading image 20060531_0312_c2.gif
2007-09-12 02:33:49,924 INFO loading image 20060531_0324_c2.gif
2007-09-12 02:33:50,284 INFO loading image 20060531_0336_c2.gif
2007-09-12 02:33:50,641 INFO starting images subtractions
2007-09-12 02:33:51,049 INFO starting image blurring with sigma=0.7
2007-09-12 02:33:52,238 INFO starting objects detection
2007-09-12 02:33:56,806 INFO objects found in image1: 303
2007-09-12 02:33:56,806 INFO objects found in image2: 301
2007-09-12 02:33:56,806 INFO objects found in image3: 319
2007-09-12 02:33:56,806 INFO objects found in image4: 313
2007-09-12 02:33:56,807 INFO starting objects parameters detection
2007-09-12 02:33:59,879 INFO starting paths detection
2007-09-12 02:34:00,125 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:34:00,126 INFO [694 697 698 699 181 199 208 217]
2007-09-12 02:34:09,541 INFO loading image 20060531_0312_c2.gif
2007-09-12 02:34:09,883 INFO loading image 20060531_0324_c2.gif
2007-09-12 02:34:10,229 INFO loading image 20060531_0336_c2.gif
```

```
2007-09-12 02:34:10,577 INFO loading image 20060531_0400_c2.gif
2007-09-12 02:34:10,921 INFO starting images subtractions
2007-09-12 02:34:11,328 INFO starting image blurring with sigma=0.7
2007-09-12 02:34:12,562 INFO starting objects detection
2007-09-12 02:34:16,850 INFO objects found in image1: 302
2007-09-12 02:34:16,850 INFO objects found in image2: 319
2007-09-12 02:34:16,851 INFO objects found in image3: 313
2007-09-12 02:34:16,851 INFO objects found in image4: 307
2007-09-12 02:34:16,851 INFO starting objects parameters detection
2007-09-12 02:34:19,976 INFO starting paths detection
2007-09-12 02:34:20,199 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:34:20,200 INFO [697 698 699 701 199 208 217 237]
2007-09-12 02:34:29,596 INFO loading image 20060531_0324_c2.gif
2007-09-12 02:34:29,972 INFO loading image 20060531_0336_c2.gif
2007-09-12 02:34:30,317 INFO loading image 20060531_0400_c2.gif
2007-09-12 02:34:30,669 INFO loading image 20060531_0412_c2.gif
2007-09-12 02:34:31,012 INFO starting images subtractions
2007-09-12 02:34:31,407 INFO starting image blurring with sigma=0.7
2007-09-12 02:34:32,640 INFO starting objects detection
2007-09-12 02:34:36,758 INFO objects found in image1: 305
2007-09-12 02:34:36,758 INFO objects found in image2: 313
2007-09-12 02:34:36,758 INFO objects found in image3: 307
2007-09-12 02:34:36,758 INFO objects found in image4: 320
2007-09-12 02:34:36,759 INFO starting objects parameters detection
2007-09-12 02:34:39,874 INFO starting paths detection
2007-09-12 02:34:40,099 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:34:40,100 INFO [698 699 701 702 208 217 237 246]
--
2007-09-12 02:57:06,535 INFO starting objects detection
2007-09-12 02:57:10,855 INFO objects found in image1: 337
2007-09-12 02:57:10,856 INFO objects found in image2: 303
2007-09-12 02:57:10,856 INFO objects found in image3: 311
2007-09-12 02:57:10,856 INFO objects found in image4: 315
2007-09-12 02:57:10,856 INFO starting objects parameters detection
2007-09-12 02:57:13,974 INFO starting paths detection
2007-09-12 02:57:15,539 INFO loading image 20060601_2136_c2.gif
2007-09-12 02:57:15,894 INFO loading image 20060601_2148_c2.gif
2007-09-12 02:57:16,234 INFO loading image 20060601_2200_c2.gif
2007-09-12 02:57:16,577 INFO loading image 20060601_2212_c2.gif
2007-09-12 02:57:16,916 INFO starting images subtractions
2007-09-12 02:57:17,311 INFO starting image blurring with sigma=0.7
2007-09-12 02:57:18,555 INFO starting objects detection
2007-09-12 02:57:22,797 INFO objects found in image1: 291
2007-09-12 02:57:22,797 INFO objects found in image2: 311
```

```
2007-09-12 02:57:22,797 INFO objects found in image3: 315
2007-09-12 02:57:22,798 INFO objects found in image4: 295
2007-09-12 02:57:22,798 INFO starting objects parameters detection
2007-09-12 02:57:25,882 INFO starting paths detection
2007-09-12 02:57:26,092 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:57:26,093 INFO [673 675 676 677   7  15  22  31]
2007-09-12 02:57:35,259 INFO loading image 20060601_2148_c2.gif
2007-09-12 02:57:35,603 INFO loading image 20060601_2200_c2.gif
2007-09-12 02:57:35,953 INFO loading image 20060601_2212_c2.gif
2007-09-12 02:57:36,301 INFO loading image 20060601_2224_c2.gif
2007-09-12 02:57:36,651 INFO starting images subtractions
2007-09-12 02:57:37,067 INFO starting image blurring with sigma=0.7
2007-09-12 02:57:38,258 INFO starting objects detection
2007-09-12 02:57:42,635 INFO objects found in image1: 318
2007-09-12 02:57:42,636 INFO objects found in image2: 315
2007-09-12 02:57:42,636 INFO objects found in image3: 295
2007-09-12 02:57:42,636 INFO objects found in image4: 325
2007-09-12 02:57:42,636 INFO starting objects parameters detection
2007-09-12 02:57:45,814 INFO starting paths detection
2007-09-12 02:57:46,035 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 02:57:46,036 INFO [675 676 677 679  15  22  31  39]
--
2007-09-12 20:03:00,888 INFO starting objects detection
2007-09-12 20:03:05,296 INFO objects found in image1: 386
2007-09-12 20:03:05,296 INFO objects found in image2: 437
2007-09-12 20:03:05,296 INFO objects found in image3: 393
2007-09-12 20:03:05,296 INFO objects found in image4: 365
2007-09-12 20:03:05,297 INFO starting objects parameters detection
2007-09-12 20:03:09,457 INFO starting paths detection
2007-09-12 20:03:11,325 INFO loading image 20061212_0836_c2.gif
2007-09-12 20:03:11,675 INFO loading image 20061212_0900_c2.gif
2007-09-12 20:03:12,024 INFO loading image 20061212_0912_c2.gif
2007-09-12 20:03:12,378 INFO loading image 20061212_0936_c2.gif
2007-09-12 20:03:12,727 INFO starting images subtractions
2007-09-12 20:03:13,137 INFO starting image blurring with sigma=0.7
2007-09-12 20:03:14,471 INFO starting objects detection
2007-09-12 20:03:18,547 INFO objects found in image1: 420
2007-09-12 20:03:18,548 INFO objects found in image2: 393
2007-09-12 20:03:18,548 INFO objects found in image3: 365
2007-09-12 20:03:18,548 INFO objects found in image4: 431
2007-09-12 20:03:18,548 INFO starting objects parameters detection
2007-09-12 20:03:22,717 INFO starting paths detection
2007-09-12 20:03:23,193 INFO possible path: [x1 x2 x3 x4 y1 y2 y3 y4]
2007-09-12 20:03:23,195 INFO [188 190 191 194  43  63  73  93]
```

Bibliography

[1] Filip Hroch. The robust detection of stars on ccd images. In *Experimental Astronomy 9*. Kluwer Academic Publishers, 1999.

[2] Paul F. Whelan Kevin Robinson. Efficient morphological reconstruction: a downhill filter. In *Pattern Recognition Letters 25*, pages 1759–1767. 2004.

[3] M. K.Hu. Visual pattern recognition by moment invariants. In *IEEE Transactions on Information Theory*, pages 179–187. 1962.

[4] Tony Lindeberg. Scale-space: A framework for handling image structures at multiple scales. In *CERN School of Computing*, 1996.

[5] X. Zhuang R. M. Harlick, S. R. Stenberg. Image analysis using mathematical morphology. In *IEEE Transactions on Pattern Analysis*, pages 532–550. 1987.

[6] J. Serra. Image analysis and mathematical morphology, volume 1. 1982.

[7] J. Serra. Image analysis and mathematical morphology, volume 2. 1988.

[8] Luc Vincent. Morphological grayscale reconstruction: Definition, efficient algorithm and applications in image analysis. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 633–635, 1992.

Credits

A closing credits section is not, as far as I experienced so far, a normal procedure out of Italy, but for once I would like to stick to the costumes of my country and dedicate a few lines to some very important people.

Italian
Il primo ringraziamento va ovviamente a mia madre, senza la quale non potrei avrei mai potuto godere dei privilegi che ho e della gioia nel vedere una mia idea prendere forma e vita.

Un ringraziamento speciale va a Zita, Silvio e Romeo, che non ho mai sentito lontani e che sono per me una famiglia.

E poi tutti quelli che mi sono venuti a trovare dall'Italia in questi due anni. In ordine di apparizione: Lollo, Scila, Elisa e Ricky. Grazie per essere passati dal dire al fare.

English
In my two years long stay here in Denrmak i had the opportunity and pleasure to meet a lot incredible people, way too many to name, but a few of them really deserve to be mentioned.

Arnaud, to be a sincere and loyal friend. Claudio, for all the times you made me laugh and for teaching me incredibol proper inglis. Marcos, for the crazy time spent together. Emiliano, for the theories that we developed together and that would probably deserve to be published. Trond, for being a wonderful person and for the effort to keep in touch regardless of distance. Rafnar, to be a brilliant guy and great friend. Vidar, for the Italian lessons i gave you, both

regarding language and how to fight in a bush of stinging nettle.

Danish (well... more or less)
Tusind tak til Ayob og Sennait, min familie i Danmark. I dag kunne jeg ikke være her uden jeres hjælp!

Mine naboer! Jeg har haft sikke in god tid i Nybrogård! Bjarne og Mads, jeg har så mange gode hukommelser sammen med jer! Anders, Johannes, lille Anders, Johannes, Dennis: i styrer! :-) Malene, Line, Nanna, Mia, Charlotte: i behræfter stereotyper om danske kvinder, men i er såmeget mere end smykke piger! Jeg føler så heldig for at kende jer!