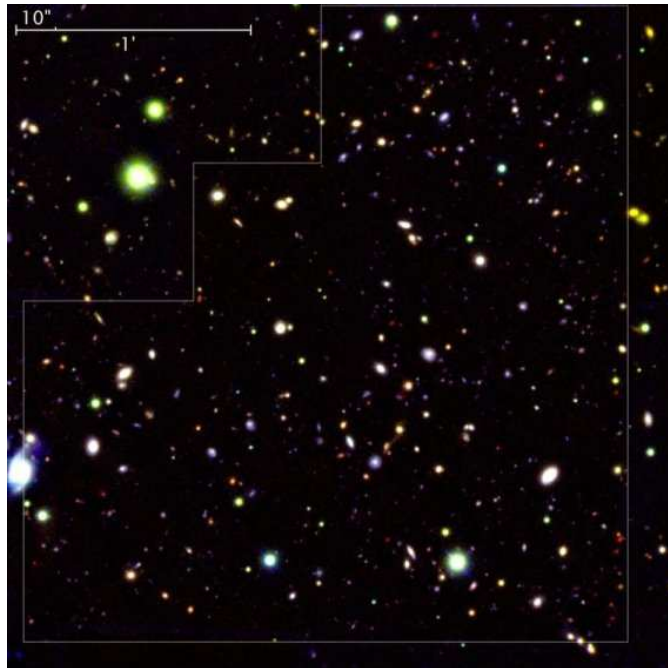


SEARCH FOR $z \sim 2$ GALAXIES IN THE HUBBLE DEEP FIELD SOUTH



B.Sc. thesis at the Niels Bohr Institute,
University of Copenhagen.
Advisor: Johan P. U. Fynbo,
DARK Cosmology Center

Thøger Juul Thorsen
thoeger@fys.ku.dk
CPR: [REDACTED]

Submitted: June 2, 2006

*Front page illustration: Composite Image of the Hubble Deep Field South,
found at <http://www.astro.helsinki.fi/~mjuvela/>*

Resumé

En af kosmologiens hovedudfordringer i denne tid er kortlægningen og beskrivelsen af det mørke stof, der udgør langt størstedelen af Universets samlede masse. Mørkt stof vekselvirker ikke med elektromagnetisk stråling, men gravitationelt vekselvirker det på normal vis med almindeligt, ueksotisk stof. Derfor, og ud fra nogle betragtninger om Universets homogenitet og isotropi, må de største ansamlinger af mørkt stof være at finde, hvor også de største ansamlinger af almindeligt stof er, hvorfor fordelingen af galakser kan fungere som indikator for det mørke stofs fordeling. Desuden kan størrelsen af de tidlige galakser ved høj rødforskydning fortælle en del om temperaturen af det mørke stof i Universet. En høj temperatur er jo som bekendt det samme som en stor mængde tilfældig molekylbevægelse, så en høj temperatur i det mørke stof vil resultere i en jævnere stoffordeling i Universet og dermed mindre ansamlinger, der er tætte nok til at afføde galaksedannelse. Det er derfor af interesse ikke blot at finde stadig fjernere objekter, men også at kortlægge det lidt nærmere univers så præcist som muligt, og til dette formål er det nødvendigt at lede efter mere lyssvage objekter.

I dette bachelorprojekt beskriver og anvender jeg en relativt ny metode til at finde lyssvage galakser ved høj rødforskydning. Metoden udnytter den snævre, men skarpe Lyman α -emissionslinje - udsendt med en bølgelængde på $\lambda = 1215.7 \text{ \AA}$ ved rødforskydning $z = 2.14$ og siden rødforskydning til en bølgelængde på omkring $\lambda = 3800 \text{ \AA}$ - til dels at detektere, dels at afstandsbestemme disse lyssvage galakser ved at sammenligne et billede taget i et meget smalt filter med et billede taget i et betragteligt bredere filter. Objekter ved en rodforskydning - og dermed afstand - der bringer deres $Ly\alpha$ -linje ind i filteret centreret omkring de ca 3800 \AA , vil forekomme betragteligt stærkere i det smalle filter end i det brede, hvorfor en simpel farve-farve-fotometri kan afsløre dem.

Til dette formål har jeg anvendt allerede eksisterende arkivdata: et smalfilterbillede taget i Chile med *Very Large Telescope*, VLT, tilhørende European Southern Observatory, samt et bredt UV-filterbillede taget på *Cerro Tololo Inter-american Observatory*, CTIO, ligeledes i Chile. Fra disse billeder er alle lyskilder, der med sikkerhed kan skelnes fra støj, ved hjælp af programmet *SExtractor* lokaliseret, opskrevet på liste og deres lysstyrke målt. Derefter er hver indgang på listerne via IDL-programmet *Combinecats*, skrevet af undertegnede, parret sammen med den indgang i listen fra det andet billede, som svarer til det samme objekt på himlen. Herefter er de, ligeledes vha *Combinecats*, blevet sammenlignet i lysstyrke, og de aktuelle objekter herved sorteret fra og vurderet nærmere.

Undersøgelsen resulterede i fundet af en række kandidatgalakser, hvoraf enkelte er vist og diskuteret.

Contents

1	Preface	1
1.1	The Hubble Sphere	1
1.2	The Formation of Structure	2
1.3	So, what?	3
2	Tools	5
2.1	Ly α lines	5
2.2	Color-color photometry	5
3	The Task	8
3.1	Intermezzo: The Astrophysical Virtual Observatory	8
3.2	The Telescopes and the Images	10
4	The Strategy	11
5	The Process	12
5.1	Preparations	13
5.2	Photometry	13
5.3	Coupling	14
5.3.1	Twins	16
5.3.2	Virtual Distance Tolerance	16
5.4	Color-color photometry	17
5.5	Gathering the orphans	18
6	Results	19
6.1	Twins	19
6.2	Orphans	21
7	Discussion and Future Work	25
8	Aknowledgements	27

A	Additional figures	29
A.1	Adjustment of reference pixels	30
A.2	Examples of different SExtractor settings	31
A.3	Orphan candidate objects	33
A.3.1	The 'good' group	33
A.3.2	The 'fair' group.	35
B	Source Code	36
B.1	Source Extractor configuration files	37
B.1.1	Used for the CTIO (broad-band) image	37
B.1.2	Used for the VLT (narrow-band) image	39
B.2	IDL program correcting reference pixel of FITS image	41
B.3	IDL program clipping the CTIO image	42
B.4	IDL program combining SExtractor catalogs	44

Chapter 1

Preface

Not many years ago, it was discovered that on the large scale - the scale on which even the breathtakingly large galaxies are only small building bricks - the structure of the Universe is filamentary, somewhat resembling a bubbly foam. Matter is concentrated along filaments and in the nodes connecting these, engulfing huge voids of very low matter density. The origin and evolution of this structure has become one of the main fields of research for modern cosmologists, and one of the prerequisites for learning more about the Large Scale Structure is a mapping of the galaxies as precise as possible.

A small brick in this building, the aim of this project is to detect and map galaxies with a redshift of $z = 2.14$ in the Hubble Deep Field south, using stock images from past observations done from the VLT and CTIO observatories in Chile.

1.1 The Hubble Sphere

Light has a finite, though large, propagation speed. A look out to any distance is a look back in time, to the time the light was emitted. All we can see of the infinite Universe, and all we can ever hope to see, is the interior of the so-called *Hubble Sphere*, a huge sphere with radius determined by the age of the Universe. The 'shell' of this sphere, the limit of our view, is the *Cosmic Microwave Background*, the CMB. Before the the emission of the CMB - that is, spatially beyond the CMB - the Universe was still as hot and dense as the interior of a star. High-energy photons would knock any electron captured by the electric attraction of a proton out of its orbit, rendering the entire Universe an extremely hot, opaque plasma for the first 300 000 years after the Big Bang.

At that age, the expansion had finally cooled down the Universe so much that protons could capture electrons in what is called the *era of recombination*. The Universe became transparent, and the many high-energetic photons could now move freely through the vast emptiness of Space. While

these photons have been traveling their way to our detectors, the further expansion of the Universe have redshifted ¹ these originally high-energetic photons into the Microwave wavelengths. The CMB, though today very low-energetic and certainly not worth much for cooking, is surprisingly dense: in number and overall energy, the CMB photons vastly outshine the combined light of all stars, galactic nuclei and other light-emitting objects of the entire Universe.

On our side of the CMB we see very little until we get very close to our own neighborhood and our own time, and what we see out there is only the strongest light sources; the largest and hottest galaxies, and active galaxies such as *Quasars* and *Seyfert Galaxies*. These strong light sources, we can be pretty sure, do *not* give a good picture of the average composition and behaviour of the Universe at that age [6]. The situation is sketched in fig. 1.1. To get better knowledge of the Universe, we should both search for ever farther objects and for fainter objects closer to us.

1.2 The Formation of Structure

Looking at the CMB, we see small temperature fluctuations, reflecting matter density fluctuations in the Universe at the *Time of Last Scattering*, when the CMB finally detached from matter. Recent measurements have shown that these density fluctuations were extremely small, having a 'depth', a relative difference between the 'hills' and 'valleys' of matter density, of about a puny $\delta \approx 3 \times 10^{-5}$ [6]. No physical model can explain how these extremely small differences in density could have developed into the enormous fluctuation amplitudes we see today. Only one solution seemed possible: that some hitherto unknown *Dark Matter*, DM, that interacts gravitationally but not with electromagnetic radiation, is present in large amounts. This DM would not have interacted with the high-energy photons before recombinations, giving them a considerable time to collapse into the density structures we see today before the Time of Last Scattering, rendering the overall matter density fluctuations much deeper than the visible, *baryonic* matter would suggest. Today's measurements suggest that about 90% of matter in the Universe is DM [6]. Hence, DM governs the gravitational properties of the Universe.

But since the DM gravitationally interacts normally with baryonic matter, the distribution of baryonic matter, readily indicated by light-emitting galaxies, also works as an indicator for the distribution of DM.

¹That is, stretched the light waves as if they were painted on the surface of a balloon being inflated. Longer wavelength is the same as more reddish light.

1.3 So, what?

Now, this is what we already know. What is now interesting to investigate is the details about the distribution of matter. The cosmological models contain two kinds of DM, hot and cold. The hot DM tends to smooth out structure formation by its large amount of random movement, whereas cold DM tends to strengthen it by its gravity [J. Fynbo, conversation].

Galaxies work as an indicator of the matter density distribution in the Universe. Finding out more about the distribution, size etc. of the early Universe at high redshifts also provides an opportunity to learn more about the distribution and temperature of the DM present, aiding in the modeling of the past and future of the Universe.

The field chosen for this project, the *Hubble Deep Field South* (HDF-S), was chosen for the large amount of image data covering this area available. This will allow not only the detection and mapping of these galaxies, but also the further investigation of their size and physical properties later on, allowing for a detailed modeling of these young galaxies and hence help understand the history of galaxy formation.

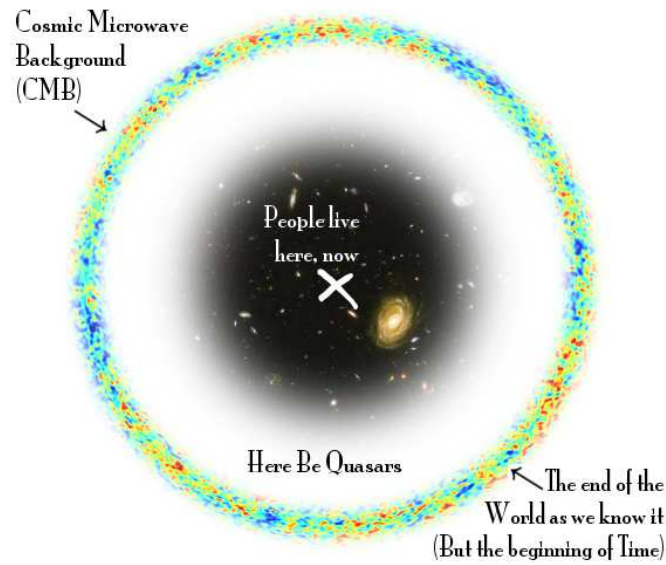


Figure 1.1: The Hubble Sphere, also known as *The Observable Universe*, sometimes misleadingly just called *The Universe*. The farther out and the further back we look, the less we see, until we reach the Cosmic Microwave Background, the limit of the observable Universe. The Hubble Sphere is the spherical part of the Universe, of which light has had time to reach us since the Big Bang. Any viewer, in any part of the Universe, will find herself (or *itself*) in the exact centre of her own Hubble Sphere. Image by the author.

Chapter 2

Tools

The two main scientific tools used in this project are the usage of the *Lyman α emission line* of neutral hydrogen, and the color-color photometry. Here follows a quick description of each.

2.1 Ly α lines

In areas of intense star formation, the strong electromagnetic radiation of the bright, young stars collide with atoms of the vast surrounding clouds of neutral hydrogen, knocking the electrons off the atom, thus ionizing the hydrogen. Later, the electrons recombine with the protons, re-emitting the high-energy photons in smaller quanta corresponding to the energies of the transitions between the different energy states in the hydrogen atom, the dominant of which is the *Lyman alpha transition* between the second-lowest and lowest energy level, corresponding to a photon wavelength of $\lambda = 1216 \text{ \AA}$. This gives the galaxy an extra strong emission at this wavelength, yielding a sharp peak in its spectrum, see figure 2.1.

This Lyman α emission line, *Ly α* , is a very useful tool in our attempts to map the galaxies and DM of the Universe at high redshifts. It is only one of many atomic spectral lines, but one of the most interesting for cosmologists, because - besides that it is the strongest emission line of hydrogen - hydrogen by far is the most common element in the Universe and even more so earlier, making it especially interesting for looking at the Universe as a whole.

2.2 Color-color photometry

After emission, the Ly α photons are continuously redshifted on their way to our telescopes, making the peak move towards the red end of our spectrum. The larger the distance, the larger the redshift and thus the wavelength of the peak. In the search for galaxies at any given distance, one needs

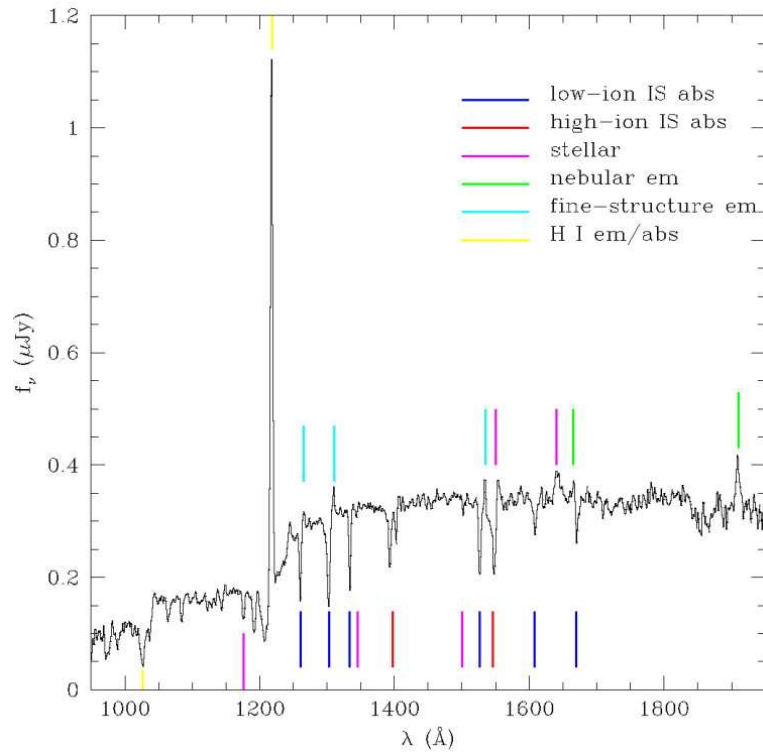


Figure 2.1: Example spectrum: the composite light of a large group of galaxies, with strong Ly α emission. The Ly α line is the sharp peak marked with yellow. The area under the graph can be interpreted as the overall flux of the object. Wavelengths shown are non-redshifted (laboratory) wavelengths.

Image from [5]

only pick the corresponding redshift¹, and look for any galaxies having their Ly α emission line falling within a narrow range around this. Observing any object in a narrow wavelength band around this peak makes an otherwise faint object seem much brighter in this filter, compared to the objects whose emission line (if any is present) falls outside the narrow filter.

The emission line, being very narrow, has no effect on the overall brightness of the object. When observed in a broader filter, the object seeming bright in the narrow filter, will appear very faint. A comparison between a narrow-band and a broad-band image now shows which objects have Ly α lines within the narrow band, telling us with good precision the redshift of these objects from our knowledge of the original wavelengths of the Lyman α transition.

All we need now is to pick a color band to look in, and thereby a redshift, and we're ready to go hunt some high- z galaxy.

¹In fact, because the Hubble Constant is not very well known, distances in the cosmological scale are most commonly measured simply by redshift

Chapter 3

The Task

In this particular project, the aim has been a closer research of the Hubble Deep Field South. This area, a roughly square patch of the sky, is situated in the southern hemisphere close to the celestial south pole, in the constellation Tucana, approximately covered by the cross-hairs at figure 3.1.

The HDF-S is centered around a smaller patch of the sky, the actual Hubble Deep Field South, a small area to which the Hubble Space Telescope, HST, was exposed for a long time, to catch a glimpse of some of the faintest objects yet seen in the Universe. The area is pointing out of the galactic disk of the Milky Way to prevent bright foreground stars and clouds of interstellar dust and gas from obscuring the view of the distant galaxies. Together with the original Hubble Deep Field, sometimes called the Hubble Deep Field North, this gives not only some stunning pictures of extremely distant objects, but also a valuable base of comparison with ground-based observations. The data for this project are archive data covering this area and its nearest surroundings, but are taken from ground-based observatories as a part of other research projects.

3.1 Intermezzo: The Astrophysical Virtual Observatory

The idea of using stock data taken by other astronomers is plain common sense. Most of the observations in modern astronomy, and especially in cosmology, requires top-notch, expensive equipment, to which we do have limited access. On the other hand, the image data already available from previous observations are far from being fully exploited. Lots of interesting research can be done based on today's stock data alone.

The large amount of stock data is not just a nice opportunity for little B.Sc students to do a bit of real research, though. The amount of data available is growing at a tremendous rate these years, even faster than the computing power available for data processing and the bandwidth for



Figure 3.1: The HDF-S position in the southern night sky. The CTIO image is about the size of the pointer ring, the original patch covered by the HST a good deal smaller, and the VLT Image somewhere in between. Image made by the author with the free astronomy software package Stellarium.

data transfer [8]. This situation calls for a more organised way of collecting, indexing and presenting these data. Enter the *Astrophysical Virtual Observatory*, AVO.

The AVO is an international effort to collect all available electronic data in a large database, providing simplified access to and better overview of the data. Besides, a large-scale use of GRID computer technology for online data processing is being introduced to provide astronomers the facilities to do a lot of their data processing online on the AVO supercomputers, lowering the time spent on data transferring and processing, and hopefully simplifying the navigation of the large amounts of data. It is hoped that this quantitative leap forward in our knowledge base will also radically change and improve the way the majority of astronomical research is done. In many cases, a view into real space, with the expensive equipment and tedious planning required, can be replaced by a look into the 'virtual space' of already available and now well-structured data, rendering the AVO an improvement in data search and indexing for astronomers comparable to what Google has been to data search in general.

3.2 The Telescopes and the Images

Two images were the primary data for this project. The first is an image taken with the telescope 1 of the European Southern Observatory's VLT telescope array in Chile, taken in a narrow band centered around a wavelength of $\lambda = 3800 \text{ \AA}$ in the ultraviolet part of the spectrum. The Image was taken by a Japanese/French group in a survey for high-energy, so called *Lyman continuum photons*¹, from galaxies at a redshift around 3 ([3] and [4]). As its broad-band counterpart was chosen a large image taken with the Blanco 4-metre Big Throughput Camera at the likewise Chile-based *Cerro Tololo Inter-american Observatory*, CTIO, by NASA's Goddard Space Flight Centre group (GSFC). The latter covers a considerably larger area of the sky, at a somewhat lower resolution and depth.

This faces us with some extra challenges. Ideally, the objects on the two images should be perfectly aligned in order to do comparative photometry, but this would require access to one of the world's largest telescopes to have the images tailor-made for this purpose, or just a highly improbable amount of luck, none of which was available for the time being. A possible work-around for this would be a re-sizing of the image of highest resolution, rebinning the photon counts of the image pixels to new bin sizes. Software routines to achieve this are readily available. However, this would give an unwanted loss of information that is preferably avoided. As described in the following, a different path was chosen.

¹That is, Photons energetic enough to ionize neutral hydrogen, $\lambda < 912 \text{ \AA}$

Chapter 4

The Strategy

The standard data format for astronomical data is the FITS¹ image. This image format contains as a minimum an array of binary data and an ASCII header containing various information, e.g. information on camera, shutter time and shape, seeing etc., and for the images at hand: information needed for conversion of the image's (x, y) pixel coordinates to absolute celestial coordinates, measured in right ascension and declination (α, δ) .

With the celestial coordinates being the best frame of reference in this case, the strategy chosen was: instead of resizing and rebinning the images and thus losing information and precision; do the photometry on the original images, list the objects with their (x, y) coordinates of each image, then convert the coordinates into the common frame of reference, the (α, δ) coordinates, thus pairing all objects that were found in the images, and nicely list those that were only found in the narrow-band image. From here, it should - in theory - be an easy task to compare the photometric photon counts of any object in each of the two filters to see if any of the objects showed an excess emission in the narrow band. These would be the ones we were searching for.

The advantages of this strategy are clear:

- No loss of data
- Very tolerant regarding the position, depth and resolution of the images.
- The images don't have to overlap perfectly. In fact, though a trivial case, they can cover two entirely different patches of the sky. The method will be of no use, then, but it will not break down.

During the process it was also revealed that the chosen method has got some drawbacks, including that it is not reliable in crowded areas of the sky. That will not be a problem here, though.

¹short for *Flexible Image Transport System*

Chapter 5

The Process

The method, as described above, seems simple enough. But as is commonly known: when simple and beautiful theory encounters the real world, unforeseen trouble in unforeseen amounts ensue. As earlier mentioned, the two images were taken at different resolutions, depth and covering different patches of the sky. Furthermore, they are calibrated differently regarding celestial- to- image coordinate conversion, and if one had hoped that **SExtractor** would list the images in the same order, one would become disappointed. That is, coupling the objects in the two images was by no means trivial, and required a bit of work.

Most of the process described from here was done in the same IDL¹ program, the source code of which is provided in Appendix A. The workflow automated in this program was of course developed through many iterations and by trial-and-error. Several ideas that seemed good at the time they came up have made their way into and out of the program, as the number of different versions of the program backed up on my hard drive will testify.

The workflow in its ideal form can roughly be sketched as follows:

- Calibration of the reference pixels determining the $(\alpha, \delta) \leftrightarrow (x, y)$ coordinates.
- Clipping the images to roughly cover the same patch of the sky ²
- Photometry & listing
- Coupling of the list entries matching the same object.
- Plotting the magnitude differences

¹Short for *Interactive Data Language*, a computation environment and programming language especially suited for processing large data arrays.

²As mentioned above not necessary, but convenient to streamline the computings

- Some objects (from now on referred to as *orphans*) were seen in the narrow-band image, but not the broad-band image; take a closer look at them

Of course, this is not the way it actually happened, but let's pretend for some time that it was, and take the practical problems as they come. To describe the actual way it went, we'd need quite a bit of BASIC code with lots of "IF... THEN GOTO" statements.

5.1 Preparations

The first step necessary was a calibration of the coordinate system of the pictures. It showed that the narrow-band image was calibrated wrong, moving the objects several pixels away from their actual position as expressed in celestial (α, δ) coordinates. The calibration was by aid of the ESO³ USNO⁴ catalog of standard objects superimposed on the image viewed in the FITS image viewer *skycat*. The header of the FITS image was then manually corrected using the IDL script `updateref.pro`, included in Appendix B of this report. This, of course, was only a rough first step, but sufficient; the fine-tuning is to follow later. A before-and-after figure is included in appendix A.

Second step was to clip the very large broad-band image to trim away the excess fat. This is, as earlier mentioned, not strictly necessary, but it is very convenient to have the two images cover approximately the same area of the sky, to avoid lots of unnecessary computations. This was done by reading the coordinates of the four corners of the narrow-band image, converting those to pixel coordinates of the broad-band image, copying the part of the image inside those to a new image with the IDL script `clipctios.pro`.

5.2 Photometry

Next step is the photometry, done with *SExtractor*. A description of how this piece of software works is given in the announcement [1], official manual [2] or the somewhat more user-friendly handbook [7], so I'll only briefly describe it here. *SExtractor* starts by measuring the photon count of each pixel, considering any pixel below a user-given (or default) threshold value background. It then splits up the non-background area in different objects, depending on different parameters that can all be controlled by the user in the configuration file, that can be adjusted for each object. Some of the more important knobs to turn in this file include the already mentioned `THRESHOLD`, the `DEBLEND` parameter that determines how deep a drop

³Short for *European Southern Observatory*

⁴Short for *United States Naval Observatory*

in luminosity is required before a blob of light is considered two separate objects instead of one; and finally the MINAREA parameter that determines how many connected pixels above the threshold are required to distinguish an object from background noise.

What are reasonable values of these parameters depends on the image data and is to a wide extent a matter of estimating and looking at the results from different experiments. For this purpose, SExtractor outputs a *checkimage* to help determine what looks like reasonable values. Various types of checkimages can be chosen from one's needs, here only the APERTURES type is used. In appendix A, some checkimages with various SExtractor knobs turned are shown to illustrate the differences.

Besides the checkimage, SExtractor also outputs a catalog of the detected objects, their pixel coordinates, photon count, uncertainty and a column of *extraction flags*, number codes for various remarks regarding the detection and extraction of the object. These catalog files are object for the next efforts.

5.3 Coupling

The data manipulations of the following are all done within the IDL program `combinecats.pro`, included in Appendix B, unless otherwise mentioned.

Now, we've got two catalogs containing roughly, but far from exactly, the same objects. The narrow-band image is deeper than the broad-band image, and besides that we are *hoping* to find objects in the former that are not detected in the latter. On the other hand, the broad-band image turned out to have quite a lot of random noise in its upper third, yielding a number of false detections. How does one tell which entries in the catalogs refer to the same object in the sky? Sorting by celestial coordinates is the way to go, though it is not as simple as it might seem. The first problem is that the images are not calibrated to fit perfectly, due to the different resolutions and slightly different rotational orientation of the images, which makes a direct matching of coordinates very hard.

A second, more subtle, problem is that SExtractor is never better than the quality of the data it handles. Bright, clearly distinct features like foreground stars are easily found and placed with good precision; but for faint, fuzzy objects like the galaxies we are aiming for here, the border of the object and, thereby the position of the center, is not easily determined, as is illustrated in figure 5.1.

The solution to these misalignment problems has by far been the most difficult and time-consuming part of this project. What was done was the following:

- Every entry in the broad-band catalog had its pixel coordinates converted to celestial coordinates and from these to the corresponding

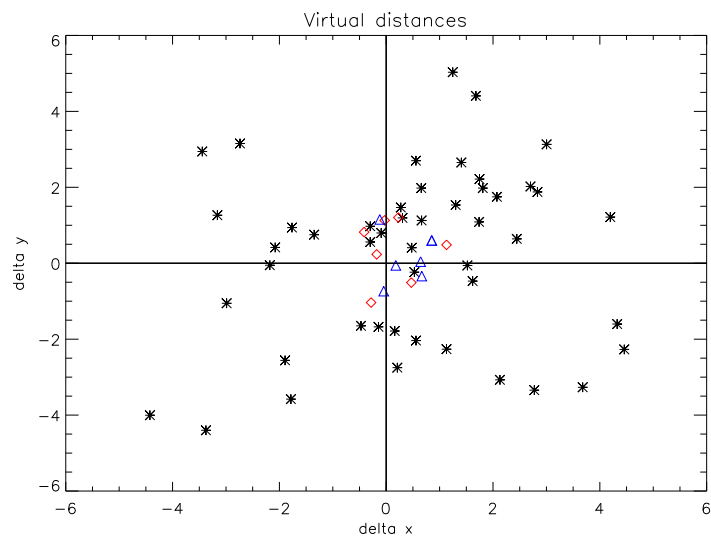


Figure 5.1: Virtual distances in the x - and y -directions of the 48 faintest objects (plotted with black asterisks) superimposed with virtual distances of 7 very bright objects (red diamonds) and 6 randomly chosen intermediate objects (blue triangles). As seen, the centers of the faintest objects are those hardest determined.

pixel coordinates of the *narrow-band* image. This should make a 'ghost' object in the narrow-band image, that should in principle be located exactly on top of the corresponding narrow-band object, but due to the above mentioned errors had a distance to it. A distance that we, short of better terms, shall call the *virtual distance* of the object.

- For every narrow-band entry, the virtual distance to all broad-band entries is now measured. The entry with the smallest virtual distance is taken as corresponding to the same objects, though it still needs double-checking.
- The objects that are detected in the narrow-band image only, the orphans, are listed for later examining. These are sorted out by choosing a smallest virtual distance we'll accept to feel assured that we actually do have two different objects here, not just some ambiguity due to a crowded area of the field. This tolerance is chosen by estimate, since we'll double-check our objects later.

The orphans are actually those of greatest interest, but for now we'll concentrate on the rest; although there is a larger chance of finding some of the desired galaxies among the orphans, objects of the desired kind spotted in both images (let's call them *twins*) will be brighter and easier to take a

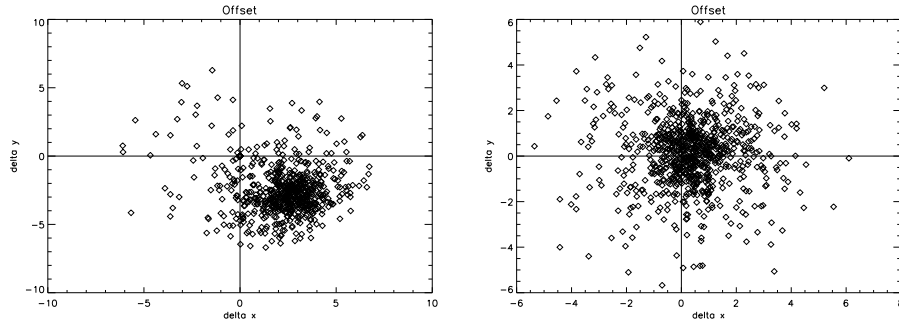


Figure 5.2: Plot of virtual distances before and after fine-tuning of the reference pixel of the narrow-band image. There is a strong systematic error in the first plot of about 2-3 pixels, an error reminiscent of the hand tuning.

closer look at later.

5.3.1 Twins

We are getting there, but not quite yet. We have no guarantee that the entries listed as corresponding to the same object actually are. In fact, a closer look shows that there are quite a lot of doublets; broad-band entries coupled to more than one narrow-band entry. This suggests that a further fine-tuning of the reference pixels is needed at this point. The fine-tuning is done by plotting the virtual distances in the x - and y - directions of the coupled objects. Any systematic error will show up here and can be corrected for like was done before. The plot in figure 5.2 shows a clear systematic error. It looks bad, but it is in fact only a couple of pixels in each direction, which isn't much, considering the high resolution of the narrow-band image, the coordinates of which are used to express the virtual distance. The error is now corrected, and in fact we have to start over from the SExtractor photometry by now - the first of our `IF... THEN GOTO` statements. The second plot in figure 5.2 shows the virtual distances after a correction and repetition of the procedure so far.

5.3.2 Virtual Distance Tolerance

Well through the next iteration, the number of doublets has dropped to only a very few as hoped, but we still have a couple of things to consider. First, the virtual distances of the twins cover a wide range. How does one choose what distance to put as the maximum tolerated before we, again, feel assured that we have the same object? To get an idea about what is sensible, a histogram of the virtual distances of the twins is plotted, shown in fig. 5.3.

The histogram shows that only a few twins are lost by choosing a value

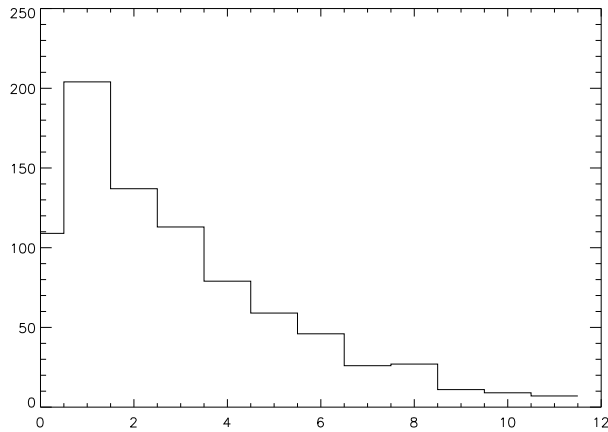


Figure 5.3: Histogram showing the density of the virtual distances.

around 6, which is in fact very close to the value used through the rest of the process. Blindly relying on a histogram like this would be a bad idea, though, the resolution of the image must also be taken into account along with the crowdedness of the field in question. Of course, a more crowded field will leave us a smaller tolerance within which we can still feel confident that the coupled entries do in fact represent the same object. The histogram should be combined with a look at the SExtractor checkimages to get an idea of the typical pixel distance between objects in the more crowded areas. Even so, one should still be cautious, all results should be checked manually to see if they're located in very crowded areas. Again, the whole procedure is repeated with this new, better tolerance value of the virtual distance.

5.4 Color-color photometry

Now that we are confident the catalog entries are coupled correctly, the comparing of fluxes can begin. The apparent magnitudes of the twins in the different filters is found by the expression for the *color index* of a light source:

$$m_U - m_N = -2,5 \cdot \log_{10} \left(\frac{f_N}{f_U} \right), \quad (5.1)$$

with m_U, f_U, m_N, f_N being the apparent magnitude and measured photon flux of the object in the broad and the narrow filter (or, generally, in any two filters), respectively. For the large majority of objects, we now expect to find more or less the same value, whereas for any objects with the strong Ly α emission line falling within our narrow filter, the color index will be

considerably lower, making the object fall distinctly below the horizontal, roughly bar-shaped branch of the others. As a little bonus, we might also see objects having an absorption line in the same narrow band.

5.5 Gathering the orphans

The twins well taken care of, it is time to return to the orphans. A priori one should think there would be more objects in the broad-band image, but in this case, this is not as deep as the narrow-band image, so the situation is actually the opposite. Objects found only in the broad-band image are easily considered objects that are simply stronger in the broad band, suggesting they are not interesting anyway. Objects found in only the narrow-band image, however, are a bit harder. Are they seen here because of an excess emission in the desired band, or simply because of the greater depth of the image? As candidates were chosen those that were clearly distinct from the nearest entry in the other catalog, i.e. with $dist \geq 5px$. From this lot, the entries that are clearly visible are taken as candidates. By 'clearly visible' is understood that the ratio of signal to Root Mean Square uncertainty, the *signal/noise coefficient* or in short S/N, is:

$$\frac{1}{\sigma} = \frac{\text{FLUX_AUTO}}{\text{RMS_AUTO}} \geq 7, \quad (5.2)$$

as taken from the narrow-band SExtractor catalog.

The value 7 is also a result of some playing around and trial-and-error. Normally, a value of $\sigma^{-1} \geq 5$ would be considered good enough, but since the broad-band image is shallower than the narrow-band image, there is an increased risk of picking orphan objects, not because they actually do have their emission line within the narrow filter, but simply because the greater depth of the narrow-band image provides better detail.

So, to be cautious, a value of $\sigma^{-1} \geq 10$ was first chosen. It turned out, though, that this value was too high; playing around with different values showed that very clearly visible objects were below the limit whereas many of the orphans with a S/N above 10 turned out to be clearly visible, large objects that had been taken for one in the broad-band image, but deblended and separated in two in the narrow-band image. So, in the end the value 7 was chosen.

After this value was chosen, it was written into the program and the program re-run another time, now picking out the twins and orphans living up to our criteria, listing them and saving these lists in two data files, `candidates.dat` and `goodorphans.dat`, for closer examination.

Chapter 6

Results

To summarize: what we would expect, or rather hope, to see is a graph of objects detected in both images lying on a roughly horizontal branch with a few objects dropping distinctly below, and a number of orphans with a reasonably high S/N value, hopefully nicely separated from other objects. Real data, of course, are a bit more complicated, though it turns out they go surprisingly well with predictions.

6.1 Twins

A plot of color index vs. virtual distance is shown in figure 6.1, with error-bars. Although the main population is somewhat smeared out, there's still a clear distinction between the 'continuum fog' and the four candidate twin objects falling below, overplotted with red diamonds. The virtual distance on the x axis is not very important, but convenient. The first plot made of this sort was simply with the running catalog number on the x -axis, giving a more even distribution of the objects; but the virtual distance is a good rule-of-thumb to check whether the result seems reliable. The smaller the virtual distance, the better.

From this plot, it was decided to use the value $U - N \leq -0.8$ as the separator, outputting exactly the four objects shown in red diamonds. The only (but sufficient) reason for using that value is that the objects below are clearly separated from the main population. In table 6.1 is listed the celestial coordinates of the four objects, along with their extraction flags.

All four objects are placed near the image borders, which should trigger our suspicion; it just seems unlikely that all objects of the kind we are searching for happen to be placed near the image border by pure chance.

Object # 4 is immediately discarded because of the flag value 24, which means flags 8 + 16, meaning that the object is too near an image border and that the objects' aperture data are incomplete or corrupted, respectively.

Objects 2 and 3 are both located in the upper, noisy third of the broad-

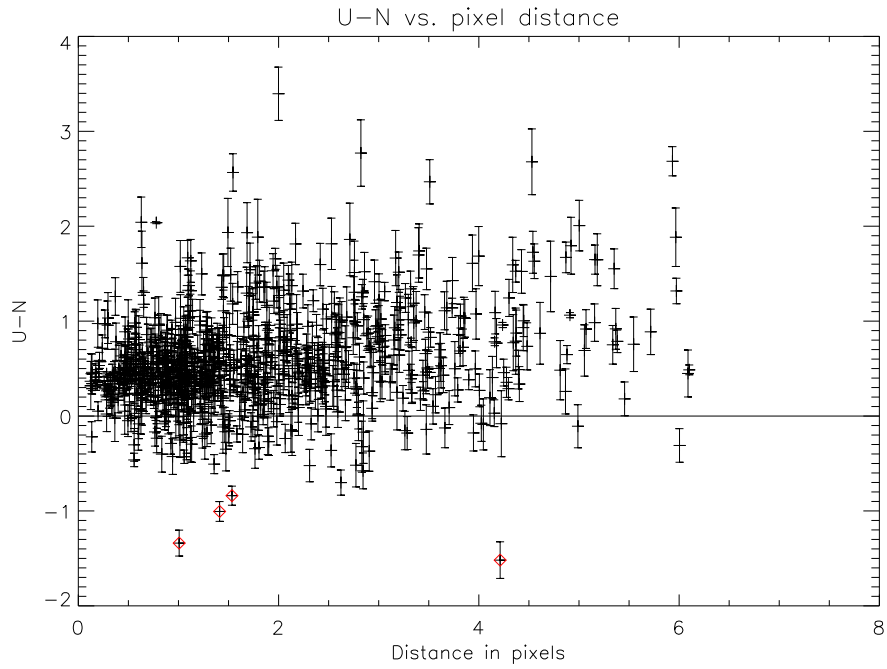


Figure 6.1: Plot of color indices with RMS errorbars of the twin objects. Note also the objects placed *above* the main group, indicating a fairly strong *absorption* in the narrow filter

Object #	(α, δ) coordinates	Extraction Flag
1	(22 : 33 : 21.670, -[60 : 36 : 11.24])	0
2	(22 : 32 : 47.106, -[60 : 29 : 58.97])	0
3	(22 : 32 : 42.674, -[60 : 29 : 58.81])	0
4	(22 : 32 : 34.524, -[60 : 32 : 10.31])	24

Table 6.1: The four candidate twin objects, their celestial coordinates and extraction flags, 0 meaning 'no comments' and 24 meaning 'too near an image border and aperture data incomplete or corrupted'.

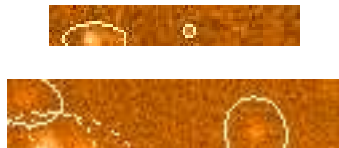


Figure 6.2: Candidate twin object, shown with larger non-candidate object (lower left) for comparison, in the broad-band and narrow-band image, respectively. The images, which are not scaled identically (in the correct scale, the horizontal distance between the two main objects would be identical), are screenshots of the SExtractor checkimages.

band image. Although they both look rather convincing, the random noise level in the upper third of the broad-band image makes additional data necessary, which is beyond the scope of this project. So, these two candidates are discarded as well.

Left is only one candidate now, # 1 in table 6.1. This is located close to the image border, too, but not close enough for SExtractor to complain about it, so let's have a closer look at it, shown in fig 6.2.

The object is the one to the right. The aperture on the narrow-band image is broken, but any missing information in the narrow-band image would add to the brightness in the narrow band, strengthening the impression that this object has indeed got a clear excess emission in the narrow band. Furthermore, the Ly α emission seems to cover a larger area (even when taking the different resolutions into account) than the ordinary UV light, fitting well with the fact that the Ly α emission is scattered much more widely in the halo of the galaxy than other wavelengths, because this halo consists mainly of neutral hydrogen, thus, according to recent simulations, giving the Ly α line a considerably increased probability of being absorbed and re-emitted, compared to other wavelengths [Peter Laursen, DARK Cosmology Center, private conversation].

However, concluding that we've caught ourselves a first-class redshift-2 galaxy would still be hubris; more data would be needed to really feel confident about it. It might be a good candidate for further investigation.

As an interesting little sidetrack, one notes that some objects on fig. 6.1 are placed a good deal *above* the main group, indicating a fairly strong absorption line in the narrow filter.

6.2 Orphans

Slightly disappointed by the results so far, though still optimistic, we now turn to the orphans, of which these with an S/N coefficient of 7 or greater are listed in the file `goodorphans.dat`. A count tells us that we have 128 of these out of 649 orphans in all. To examine these further, they were run

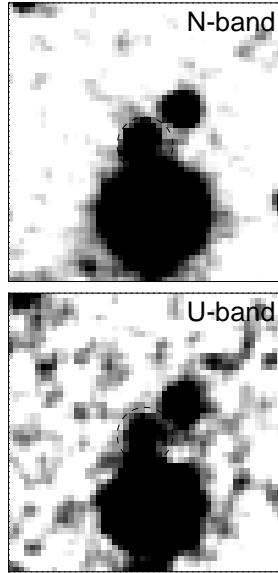


Figure 6.3: Image of orphan with high-end S/N -coefficient. The orphan (in the middle, connected to the largest object) is most likely a result of missing deblending in the U (broad-band) image.

through an IDL script written by Johan Fynbo making close-up images of an area centered at a given set of coordinates in the broad-band image and in a resized version of the narrow-band image (this is used for qualitative consideration only, so preservation of data is not very important here) and put them together for comparison. As earlier mentioned, most of the objects with high-end S/N coefficients turned out to be very bright and unusable objects, mostly very close to even brighter objects; probably a result of SExtractor deblending and separating the object in the narrow-band image but not in the broad-band image, leaving one sub-object as a 'false' orphan. An example image of this kind of objects is shown in figure 6.3

Like before, all objects on positions corresponding to the upper third of the broad-band image were immediately trashed due to the low quality of that part of the broad-band image (which can readily be seen with `skycat`). Of those left, most have a S/N between 7 and 10. The following examination consists mainly of an estimate of the difference between the images of the area around the object provided by the two filters.

We find that even in the groups of objects that looked reasonable, there is a large difference. Some are very clear in the narrow band while totally absent in the broad band, others are clearly visible in the broad-band image, though not bright enough to be detected by SExtractor. These might have been detected by setting a lower `THRESHOLD` in the configuration file, but this would also have triggered more false detections because of the random

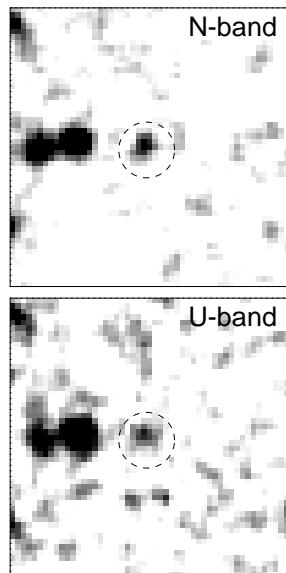


Figure 6.4: Image of an orphan object that turns out not to be so orphan after all. We cannot tell whether the excess brightness in the narrow band is due to excess emission or greater detail of the image.

noise in the image. In between these extremes, the whole range of certainty is more or less covered.

The objects were therefore examined, many were discarded, and those kept were put in groups corresponding to how well they seem to match our criteria. A *good*, a *fair* and a *bad* group were made, examples of which are shown in figures 6.4 through 6.6. In the *bad* group, the two versions of the object are so close to each other in brightness that we cannot tell whether the excess brightness in the narrow-band image is due to excess emission or just to the greater depth of the image. The *good* group is that of which we feel confident they are indeed the kind of objects we are looking for, and the *fair* group is - well, all those in between.

In all, 13 objects were counted in the best group, and four in the intermediate group, all listed in appendix A.

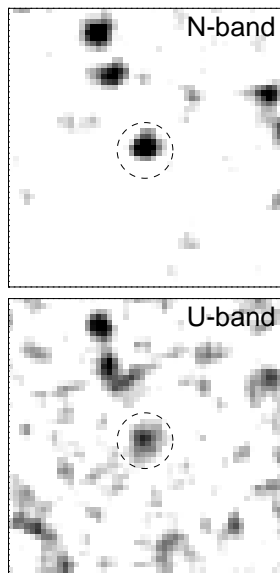


Figure 6.5: Image of an object of the intermediate group. It is pretty convincing, but not quite; the object in the broad-band image is still a bit too close in brightness.

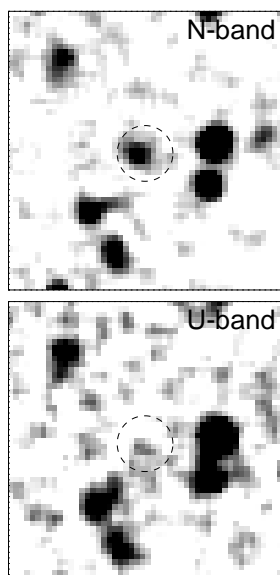


Figure 6.6: Image of an orphan objects that leaves little doubt that it is indeed an object of the wanted kind. Note that the other objects in the field are brighter in the broad band, whereas the orphan object is brighter in the narrow band, further supporting the reliability of the result.

Chapter 7

Discussion and Future Work

Although one could always wish for more, the results of this project have exceeded my expectations. One could have hoped for, maybe even expected, to find more, and more reliable, candidate objects detected in both images, but on the other hand, the number of fine orphan objects makes up for that. And although one could have wished to have delivered a piece of full-featured software ready for the market (well, maybe not), a good progress has been made towards an automated work process that can easily be used for similar tasks later.

One could also have wished for a deeper broad-band image for the comparison. The depth of the broad-band image should ideally be larger than that of the narrow-band image, not reverse, to provide the best basis for comparison of fluxes. As in fig. 6.6, seeing objects around the candidate that are clearly brighter in the broad-band image gives a warm, fuzzy feeling of confidence that an object brighter in the narrow-band image is indeed an object of the desired kind.

A deeper broad-band image, taken with the Chile-based, ESO-owned *New Technology Telescope* was in question, but the CTIO image was chosen because it covers more of the area of the narrow-band image than the NTT image does. This argument of course loses its value as it is found that the upper third of that field is unusable due to random noise. Inputting the NTT image in the `combinecats` program and including it in the comparisons would not have been very hard, were it not for the fact that some missing keywords in the FITS header of the image made some of the sub-routines choke. It might have been possible to fix this manually, but if so, it would have been too time-demanding. It would, however, be interesting to take a closer look at on a later occasion. The larger depth of the NTT image and its higher resolution might also give a better coupling of objects in the two images.

The results of the survey might also be improved by further fiddling with the SExtractor configuration files, though the values are absolutely

reasonable now.

A code cleanup of the IDL program would be another item placed high on the wish list. The program does its job, but I'm no skilled programmer, so clean and modular code that could make the program usable for others would still demand a bit working on. It would be worth it, though, a cleaner code meaning a fully automated workflow could make it a lot easier to move on to other similar tasks, performing more research faster and more efficiently.

As for the galaxies found, this project has only scratched the surface. The objects are now detected, but their location is about all we know about them. Lots of image data covering the Hubble Deep Field South in all ranges of wavelengths still lie in the online and offline archives. An interesting field of future work would be the deeper investigation of the objects found and mapped in this project; to measure their size, behavior at different wavelengths etc, to try to obtain a realistic model of their properties, hence getting a deeper knowledge of the formation of galaxies and Large Scale Structure in the early Universe.

Chapter 8

Acknowledgements

I am very grateful to the following persons for their help in different ways:

Johan Fynbo For being a patient, helpful and inspiring advisor

Kim Nilsson For invaluable help reading the report and giving highly qualified criticism.

Karen Thorsen My mother, for constructive criticism regarding language and structure of the paper.

Christina Henriksen For help with programming and general advice, and for being a nice officemate.

Christina Rivera-Beaumont For feedback on the structure and readability of the paper.

Korrie Boerm Same as above.

Bibliography

- [1] Bertin & Arnouts. SExtractor: Software for source extraction. *A & AS*, 117:393, 1996.
- [2] Bertin. SExtractor user's manual. <http://terapix.iap.fr/>.
- [3] Inoue et al. Vlt narrow-band photometry in the lyman continuum of two galaxies at $z \sim 3$. limits to the escape of ionizing flux. *A&A*, 435:471, 2005.
- [4] Iwata et al. Spectroscopy and stellar populations of star-forming galaxies at $z \sim 3$ in the hubble deep field - south. *A&A*, 440:881.
- [5] Shapley et al. Rest-frame ultraviolet spectra of $z \sim 3$ lyman break galaxies. *ApJ*, 588:65, 2003.
- [6] Johan P. U. Fynbo and Palle Møller. Galakser i det tidlige univers, i: Lyman-kanten. *KVANT*, December:22, 2000.
- [7] B. W. Holwerda. Source extractor for dummies v5. *eprint arXiv:astro-ph/0512139*, 2005.
- [8] Astrophysical Virtual Observatory. website. <http://www.euro-vo.org/avo/>.

Appendix A

Additional figures

A.1 Adjustment of reference pixels

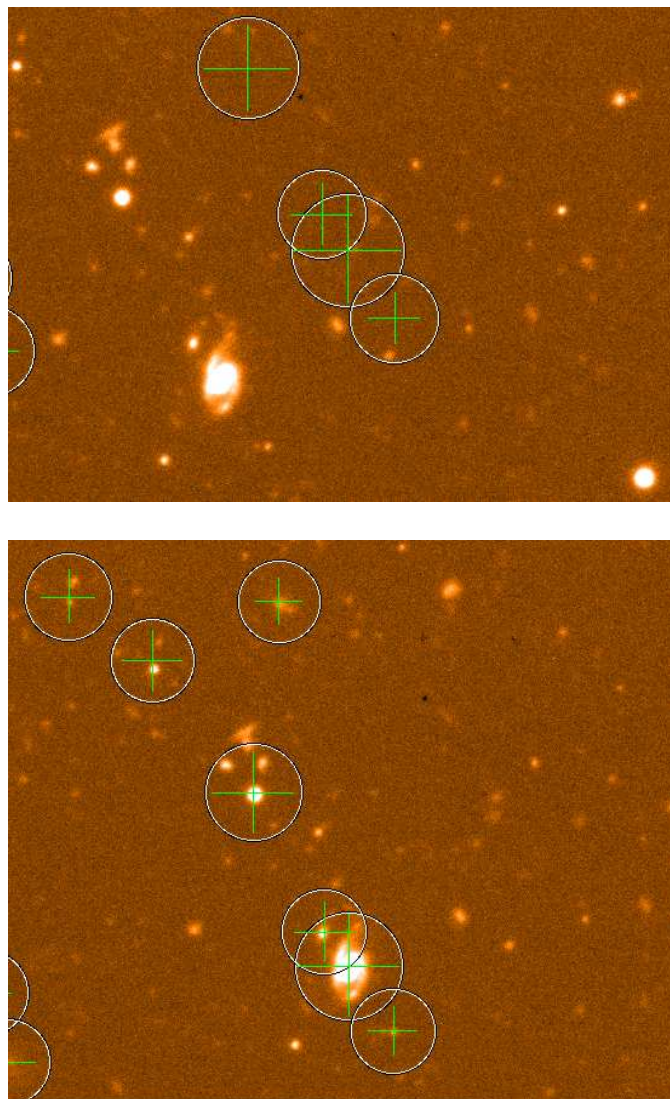


Figure A.1: The ESO USNO catalog superimposed on the narrow-band image from VLT before and after adjustment of the reference pixel.

A.2 Examples of different SExtractor settings

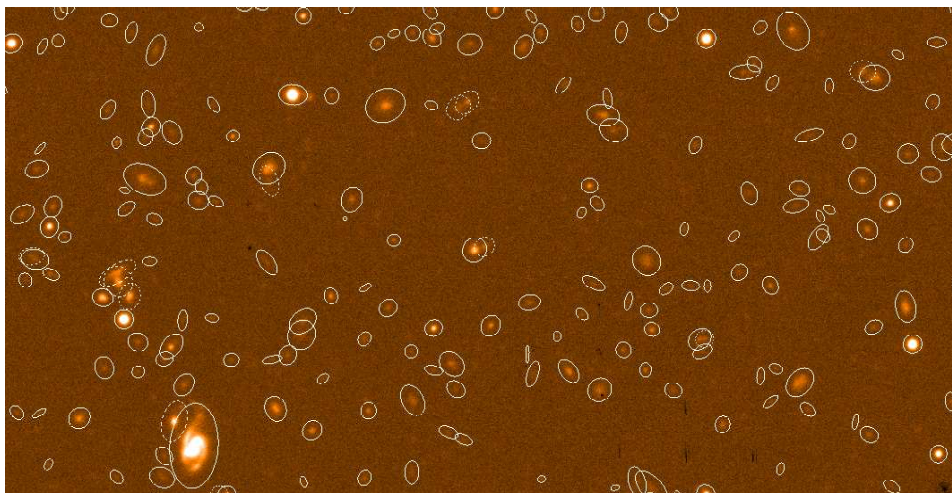


Figure A.2: Checkimage of the APERTURES kind, with reasonable values of the different parameters, of the CTIO image

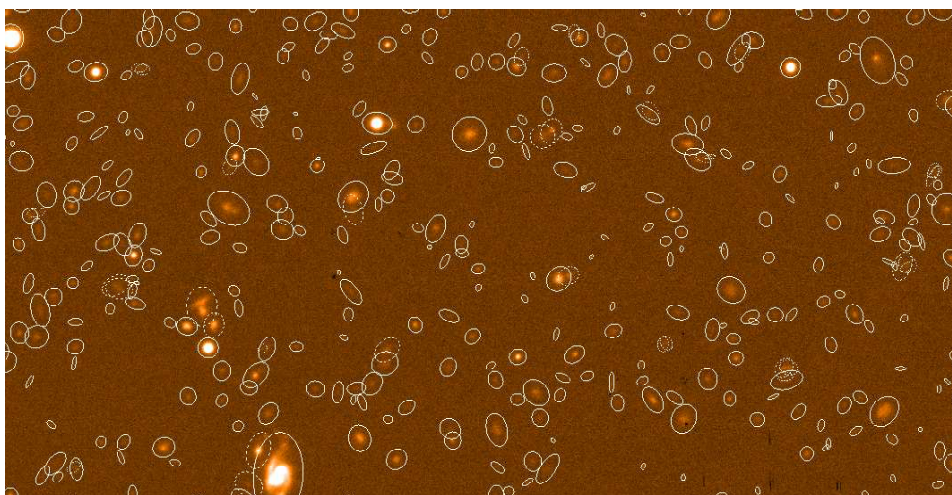


Figure A.3: Checkimage with a lower value of the THRESHOLD parameter. This will detect very small fluctuations, which cannot responsibly be distinguished from noise, as objects.

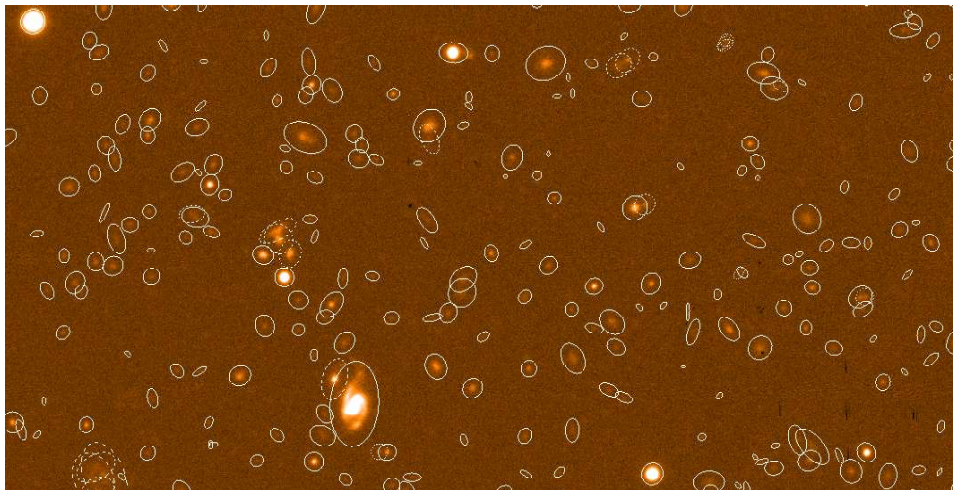


Figure A.4: Checkimage with a very small value of the MINAREA parameter, allowing two connected pixels above the threshold value to be detected as an object. This will also allow noise to be detected as objects.

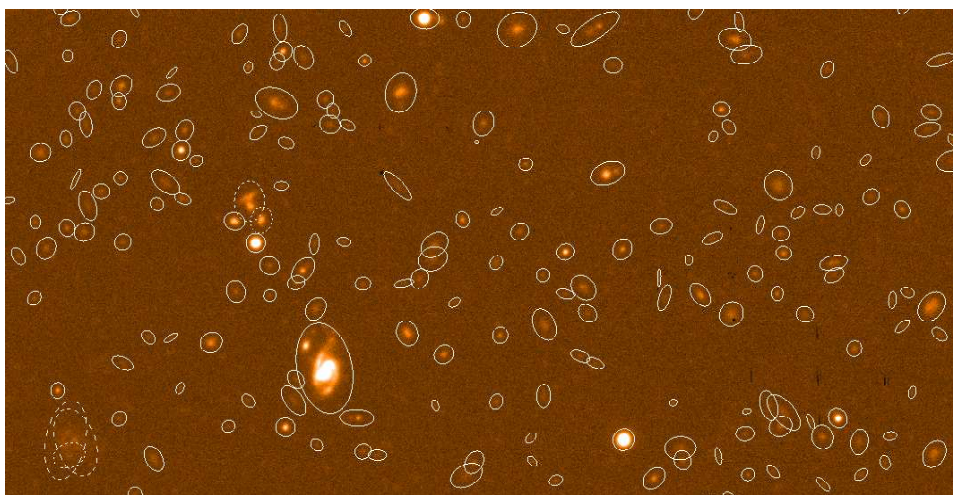


Figure A.5: Checkimage with a high DEBLENDING value. The deblending value determines when objects are considered separate objects or one object. As seen, clearly distinct objects are here taken as one.

A.3 Orphan candidate objects

A.3.1 The 'good' group

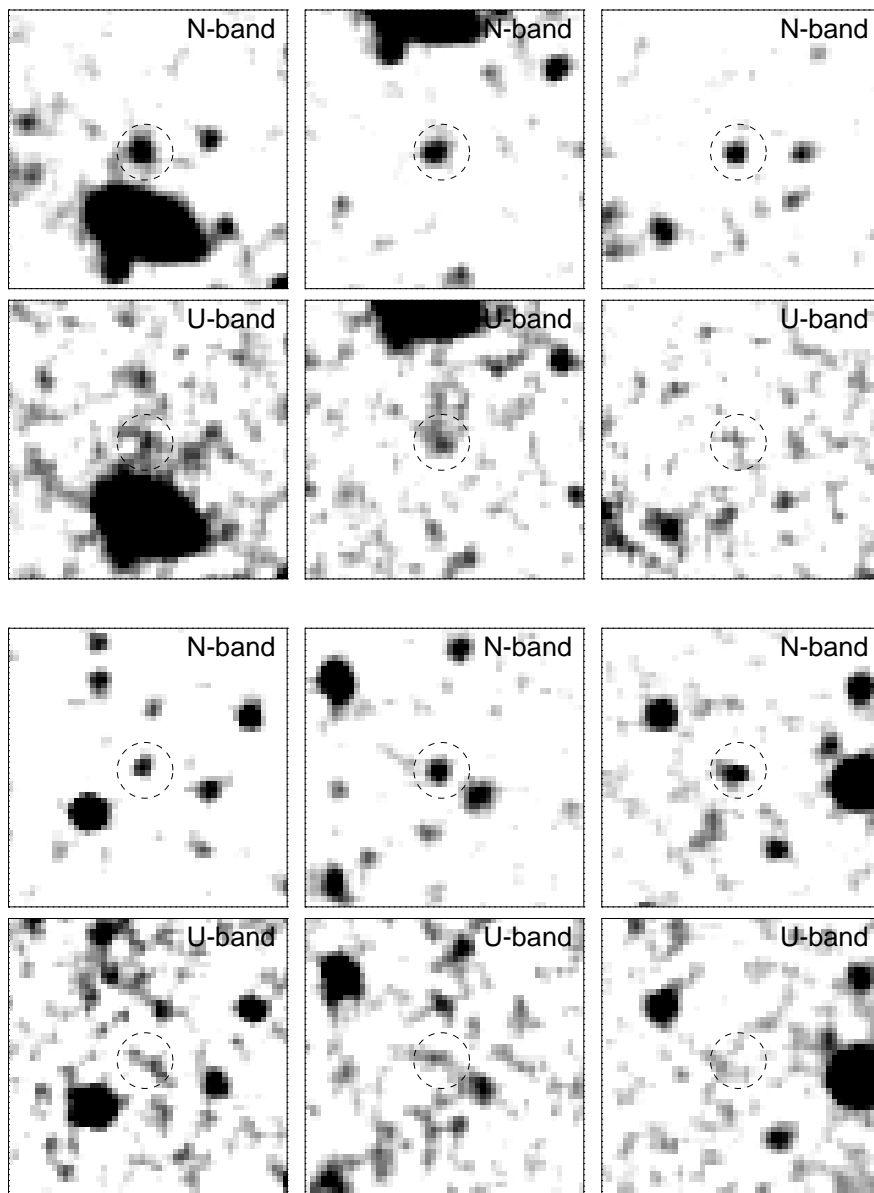


Figure A.6: Images of half of the 12 good orphan objects that are not shown in the report. Their coordinates as seen in the CTIO image are (112, 235), (116, 195), (236, 352), (395, 565), (460, 341) and (817, 237).

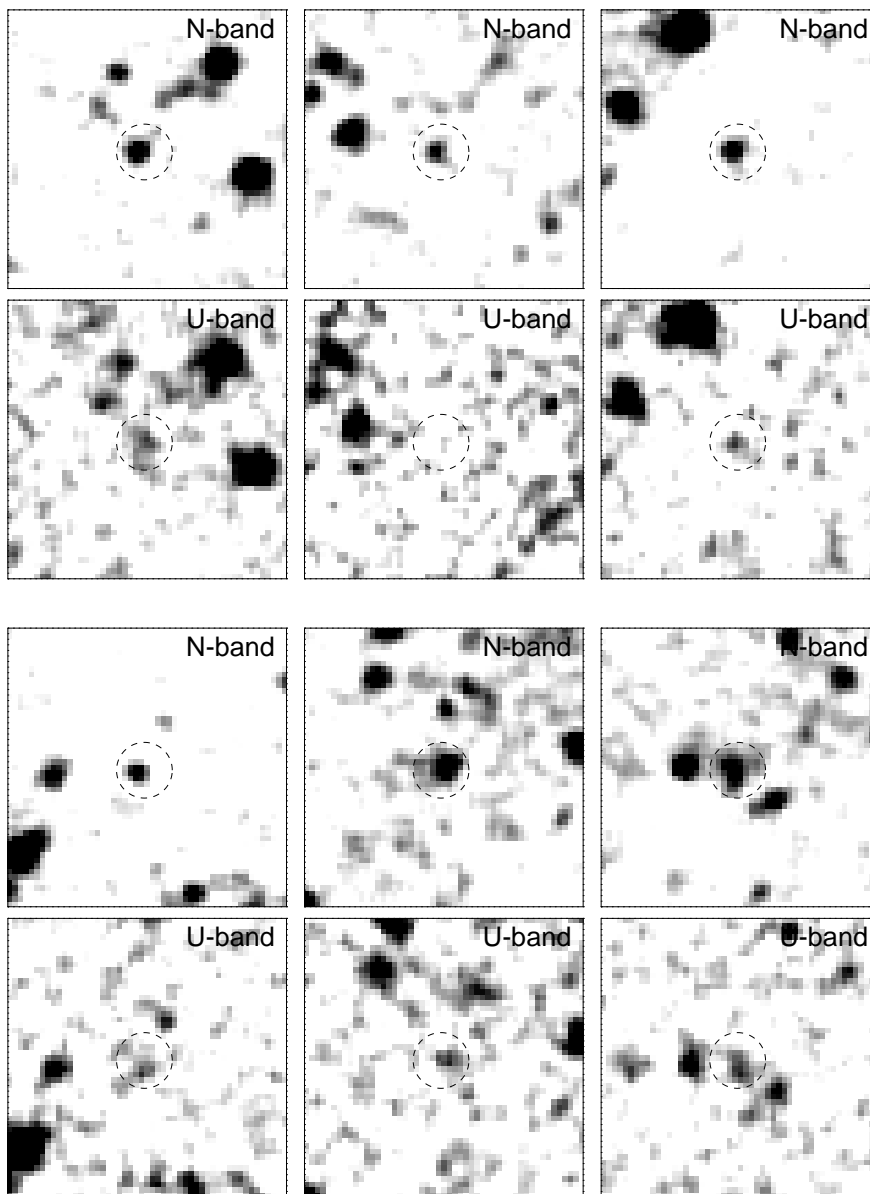


Figure A.7: Images of the other half of the 12 good orphan objects that are not shown in the report. Their coordinates as seen in the CTIO image are (62, 278), (223, 632), (697, 436), (338, 401), (652, 441) and (65, 452).

A.3.2 The 'fair' group.

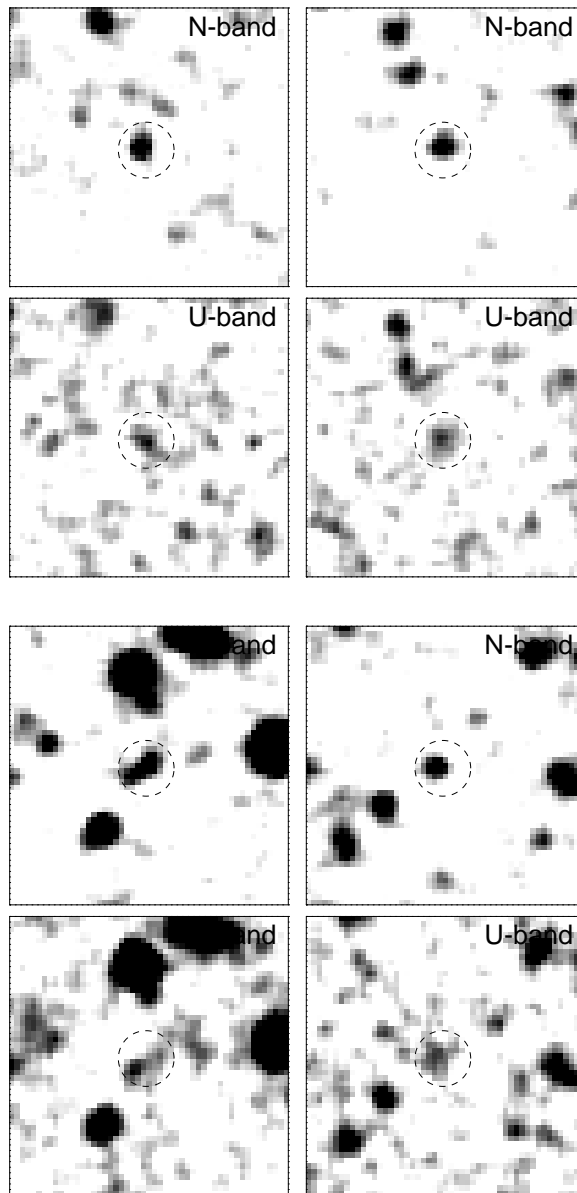


Figure A.8: Images of the four members of the 'fair' group of orphans. Their coordinates as seen in the CTIO image are (134, 443), (768, 351), (56, 352), and (222, 434).

Appendix B

Source Code

B.1 Source Extractor configuration files

B.1.1 Used for the CTIO (broad-band) image

```

# Default configuration file for SExtractor V1.2b14 - > 2.0
# EB 23/07/98
# (*) indicates parameters which can be omitted from this config file.

#----- Catalog -----
CATALOG_NAME      gsfc.cat          # name of the output catalog
CATALOG_TYPE      ASCIIHEAD        # "NONE", "ASCIIHEAD", "ASCII", "FITS_1.0"
                                      # or "FITS_LDAC"

PARAMETERS_NAME   default.param    # name of the file containing catalog contents

#----- Extraction -----
DETECT_TYPE       CCD              # "CCD" or "PHOTO" (*)
FLAG_IMAGE        flag.fits        # filename for an input FLAG-image
DETECT_MINAREA    5                # minimum number of pixels above threshold
DETECT_THRESH     1.5              # <sigmas> or <threshold>,<ZP> in mag.arcsec-2
ANALYSIS_THRESH   1.5              # <sigmas> or <threshold>,<ZP> in mag.arcsec-2

FILTER            Y                # apply filter for detection ("Y" or "N")?
FILTER_NAME       default.conv     # name of the file containing the filter

DEBLEND_NTHRESH  32                # Number of deblending sub-thresholds
DEBLEND_MINCONT   0.005            # Minimum contrast parameter for deblending

CLEAN             Y                # Clean spurious detections? (Y or N)?
CLEAN_PARAM       1.0              # Cleaning efficiency

MASK_TYPE         CORRECT          # type of detection MASKing: can be one of
                                      # "NONE", "BLANK" or "CORRECT"

#----- Photometry -----
PHOT_APERTURES    5                # MAG_APER aperture diameter(s) in pixels
PHOT_AUTOPARAMS  2.5, 3.5         # MAG_AUTO parameters: <Kron_fact>,<min_radius>

SATUR_LEVEL       50000.0          # level (in ADUs) at which arises saturation

MAG_ZEROPOINT     0.0              # magnitude zero-point
MAG_GAMMA         4.0              # gamma of emulsion (for photographic scans)
GAIN               0.0              # detector gain in e-/ADU.
PIXEL_SCALE       1.0              # size of pixel in arcsec (0=use FITS WCS info).

#----- Star/Galaxy Separation -----
SEEING_FWHM       1.2              # stellar FWHM in arcsec
STARNNW_NAME      default.mnw     # Neural-Network-Weight table filename

```

```
#----- Background -----
BACK_SIZE          64          # Background mesh: <size> or <width>,<height>
BACK_FILTERSIZE    3          # Background filter: <size> or <width>,<height>

BACKPHOTO_TYPE     GLOBAL      # can be "GLOBAL" or "LOCAL" (*)
BACKPHOTO_THICK    24          # thickness of the background LOCAL annulus (*)

#----- Check Image -----
CHECKIMAGE_TYPE    APERTURES   # can be one of "NONE", "BACKGROUND",
                                # "MINIBACKGROUND", "-BACKGROUND", "OBJECTS",
                                # "-OBJECTS", "SEGMENTATION", "APERTURES",
                                # or "FILTERED" (*)
CHECKIMAGE_NAME     gsfc-check.fits # Filename for the check-image (*)

#----- Memory (change with caution!) -----
MEMORY_OBJSTACK    2000        # number of objects in stack
MEMORY_PIXSTACK    100000      # number of pixels in stack
MEMORY_BUFSIZE     1024        # number of lines in buffer

#----- Miscellaneous -----
VERBOSE_TYPE       NORMAL      # can be "QUIET", "NORMAL" or "FULL" (*)

#----- New Stuff -----
```

B.1.2 Used for the VLT (narrow-band) image

```

# Default configuration file for SExtractor V1.2b14 - > 2.0
# EB 23/07/98
# (*) indicates parameters which can be omitted from this config file.

#----- Catalog -----
CATALOG_NAME      vlt.cat # name of the output catalog
CATALOG_TYPE      ASCII_HEAD # "NONE", "ASCII_HEAD", "ASCII", "FITS_1.0"
                                # or "FITS_LDAC"

PARAMETERS_NAME   default.param # name of the file containing catalog contents

#----- Extraction -----
DETECT_TYPE       CCD           # "CCD" or "PHOTO" (*)
FLAG_IMAGE        flag.fits     # filename for an input FLAG-image
DETECT_MINAREA    7             # minimum number of pixels above threshold
DETECT_THRESH     1.4           # <sigmas> or <threshold>,<ZP> in mag.arcsec-2
ANALYSIS_THRESH   1.4           # <sigmas> or <threshold>,<ZP> in mag.arcsec-2

FILTER            Y             # apply filter for detection ("Y" or "N")?
FILTER_NAME       default.conv   # name of the file containing the filter

DEBLEND_NTHRESH   8             # Number of deblending sub-thresholds
DEBLEND_MINCONT   0.005         # Minimum contrast parameter for deblending

CLEAN              Y             # Clean spurious detections? (Y or N)?
CLEAN_PARAM       1.0           # Cleaning efficiency

MASK_TYPE         CORRECT        # type of detection MASKing: can be one of
                                # "NONE", "BLANK" or "CORRECT"

#----- Photometry -----
PHOT_APERTURES    5             # MAG_APER aperture diameter(s) in pixels
PHOT_AUTOPARAMS   2.5, 3.5     # MAG_AUTO parameters: <Kron_fact>,<min_radius>

SATUR_LEVEL       50000.0       # level (in ADUs) at which arises saturation

MAG_ZEROPOINT     0.0           # magnitude zero-point
MAG_GAMMA         4.0           # gamma of emulsion (for photographic scans)
GAIN              0.0           # detector gain in e-/ADU.
PIXEL_SCALE       1.0           # size of pixel in arcsec (0=use FITS WCS info).

#----- Star/Galaxy Separation -----
SEEING_FWHM       1.2           # stellar FWHM in arcsec
STARNNW_NAME      default.nnw   # Neural-Network_Weight table filename

#----- Background -----
BACK_SIZE         64            # Background mesh: <size> or <width>,<height>

```

```
BACK_FILTERSIZE 3          # Background filter: <size> or <width>,<height>

BACKPHOTO_TYPE GLOBAL     # can be "GLOBAL" or "LOCAL" (*)
BACKPHOTO_THICK 24        # thickness of the background LOCAL annulus (*)

#----- Check Image -----

CHECKIMAGE_TYPE APERTURES          # can be one of "NONE", "BACKGROUND",
# "MINBACKGROUND", "-BACKGROUND", "OBJECTS",
# "-OBJECTS", "SEGMENTATION", "APERTURES",
# or "FILTERED" (*)

CHECKIMAGE_NAME vlt-check.fits    # Filename for the check-image (*)

#----- Memory (change with caution!) -----

MEMORY_OBJSTACK 2000          # number of objects in stack
MEMORY_PIXSTACK 100000       # number of pixels in stack
MEMORY_BUFSIZE 1024          # number of lines in buffer

#----- Miscellaneous -----

VERBOSE_TYPE NORMAL          # can be "QUIET", "NORMAL" or "FULL" (*)

#----- New Stuff -----
```

B.2 IDL program correcting reference pixel of FITS image

```
1  pro updateref
2
3  pic = readfits('HDFScombnew.fits', head3)
4  ;Update the header
5  refx = sxpar(head3, 'CRPIX1')
6  refy = sxpar(head3, 'CRPIX2')
7  refx = refx+2.
8  refy = refy-3.
9  sxaddpar, head3, 'CRPIX1', refx
10 sxaddpar, head3, 'CRPIX2', refy
11
12 writefits, 'adjusted-HDFScomb.fits', pic, head3
13
14
15 end
```

B.3 IDL program clipping the CTIO image

```

1  pro clipctios
2
3
4  ;picname = ''
5
6  ;::::::::::::::::::::::::::::::::::::::::::::::::::
7  ;::
8  ;:: Beskaer billederne fra CTIO, saa de passer med
9  ;:: stoerrelsen paa billedet fra VLT
10 ;::
11 ;::      —oOo—
12 ;:: ad-koordinater for det vi gerne vil beholde
13 ;::
14
15     raul = ten(15.*22.,15.*33.,15.*27.891)
16     raur = ten(15.*22.,15.*32.,15.*37.079)
17     rall = ten(15.*22.,15.*33.,15.*27.929)
18     ralr = ten(15.*22.,15.*32.,15.*37.003)
19
20     decul = -ten(60, 30, 08.19)
21     decur = -ten(60, 30, 08.25)
22     decll = -ten(60, 36, 33.08)
23     declr = -ten(60, 36, 33.14)
24
25
26 ;::::::::::::::::::::::::::::::::::::::::::::::::::
27 ;:: Indlaes liste med filer
28 ;::
29
30     openr, u, 'files.list', /get_lun
31
32 ;::::::::::::::::::::::::::::::::::::::::::::::::::
33 ;:: lav loekke der gentages lige saa mange gange som der er filer i listen
34 ;::
35
36     while not eof(u) do begin
37
38         ;::::::::::::::::::::::::::::::::::::::::::::::::::
39         ; Indlaes filnavn og billede
40         ;::::::::::::::::::::::::::::::::::::::::::::::::::
41
42         picname = strarr(1)
43
44         readf, u, picname
45         print, "Reading: ", picname
46         pic = readfits(picname, header)
47         ;hprint, header, 20
48
49         ;::::::::::::::::::::::::::::::::::::::::::::::::::
50         ; Konverter ad- til xy-koordinater
51         ;::::::::::::::::::::::::::::::::::::::::::::::::::
52
53
54         adxy, header, raul, decul, xul, yul
55         ;print, xul
56         ;print, yul
57         adxy, header, raur, decur, xur, yur
58         adxy, header, rall, decll, xll, yll
59         adxy, header, ralr, declr, xlr, ylr
60
61         ;::::::::::::::::::::::::::::::::::::::::::::::::::
62         ; Tag de gennemsnitlige af disse:
63         ;::::::::::::::::::::::::::::::::::::::::::::::::::
64
65         ;print, xul
66         ;print, xll
67
68         xl = mean([xul, xll])
69         print, 'xl = ', xl
70         xr = mean([xur, xlr])
71         print, 'xr = ', xr
72         yu = mean([yul, yur])
73         print, 'yu = ', yu
74         yl = mean([yll, ylr])
75         print, 'yl = ', yl

```

```
76
77
78     ;::::::::::::::::::::::::::::::::::
79     ; Beskaer billedet , gem beskaaret version
80     ;::::::::::::::::::::::::::::::::::
81
82     print, "Writing clipped version of", picname
83     hextract, pic, header, npic, neader, xl, xr, yl, yu, /silent
84     ;subpic = pic(xl:xr, yl:yu)
85     writefits, 'clipped-'+picname, npic, neader
86 endwhile
87
88 close, u
89 free_lun, u
90
91 ;::::::::::::::::::::::::::::::::::
92 ;;; Skal jeg aendre paa headeren eller skal jeg ikke?
93 ;::::::::::::::::::::::::::::::::::
94
95
96 end
```

B.4 IDL program combining SExtractor catalogs

```

1  pro combinecats
2
3  ;*****
4  ;** This IDL program reads two specified images in FITS format and the source catalog
5  ;** created by running them through SExtractor, and sorts the catalogs by pixel coordinate
6  ;** (luckily they can be sorted by only one coordinate), and hereafter converts the
   coordinates
7  ;** of one to celestial coordinates using the XYAD routine and further to pixel coordinates
   as
8  ;** seen in the other image using ADXY.
9  ;** The program then measures the distance between every object in one image and every object
10 ;** the other. For each object in one catalog, it compares the distances to all other objects
   ,
11 ;** and identifies the object with the one with the smaller distance, if this distance is
   less
12 ;** than a specified value.
13 ;**
14 ;** If more than one object in the first catalog identify with the same object in th other
15 ;** catalog, these entries are found and copied to another catalog for later reference.
16 ;**
17 ;** Next step is plotting the distance between the chosen objects in the X and Y direction
18 ;** see if there is any systematic error, and hereafter it plots the magnitudes of the two
19 ;** different incarnations of every object. If there is an excess luminosity in the narrow
   filter,
20 ;** this will show as a point well below the main field of objects in this plot.
21 ;**
22 ;** The objects only found in the narrowband image – 'orphans' – are listed, and those with a
23 ;** user-defined minimum virtual distance and Signal/Noise-coefficient are listed
24 ;** in a special file, 'goodorphans.dat', for later investigation.
25 ;**
26 ;** Finally – this cannot yet be user-defined in the top of the program file – it plots the
27 ;** virtual distances in the x- and y- directions for a number of bright,
28 ;** intermediate and faint objects, to illustrate SExtractor's efficiency
29 ;** defining the positions of the objects.
30 ;**
31 ;*****
32
33 ;::::::::::::::::::
34 ;; Define file names
35 ;::::::::::::::::::
36
37     pic1 = 'adjusted-HDFScomb.fits'
38     cat1 = 'vlt-nohead.cat'
39     cat1sorted = 'vlt-sorted.dat'
40     cat1sortradec = 'vlt-radec-reverse.dat'
41
42
43     pic2 = 'clipped-hdfs_btc_u.fits'
44     cat2 = 'gsfc-nohead.cat' ; edit back to gsfc-nohead.cat
45     cat2sorted = 'gsfc-sorted.dat'
46     cat2sortradec = 'gsfc-radec-reverse.dat'
47
48     ;; Tolerance for the distance within which we accept two entries as
49     ;; corresponding to the same object
50     offtouse = 6.13
51     ;; Tolerance for orphans: what should 1/sigma be?
52     tolerance = 7.
53     ;; ...and smallest accepted distance to nearest object?
54     littledist = 10.
55
56 ;::::::::::::::::::
57 ;; Read the FITS images and their headers
58 ;::::::::::::::::::
59
60     vltpic = readfits(pic1, h)
61     ctio = readfits(pic2, hdr)
62
63 ;::::::::::::::::::
64 ;; *Sort the catalogs by pixel coordinates*
65 ;::::::::::::::::::
66
67     openw, lun3, 'vlt-sorted.dat', /get_lun
68     openw, lun4, 'gsfc-sorted.dat', /get_lun
69
70     ;::::::::::::::::::
71     ;; read first catalog
72     ;::::::::::::::::::

```

```

73
74     print, 'Sorting and rearranging ', cat1
75     vlt = rdtab(cat1)
76
77     ::::::::::::::::::::
78     ::: Now sorting, column 4 = fifth col = x value.
79     ::::::::::::::::::::
80
81     sortIndex = sort( vlt[4, *] )
82
83     ::::::::::::::::::::
84     ::: The actual rearranging
85     ::::::::::::::::::::
86
87     for j = 1, 6 do vlt (j,*) = vlt (j, sortIndex)
88     printf, lun3, vlt
89     close, lun3
90     free_lun, lun3
91
92     ::::::::::::::::::::
93     ::: Repeat for the gsfc/ctio catalog
94     ::::::::::::::::::::
95
96     print, 'Sorting and rearranging ', cat2
97     gsfc = rdtab(cat2)
98     sortIndex = sort( gsfc[4, *] )
99     for j = 1, 6 do gsfc(j,*) = gsfc(j, sortIndex)
100    printf, lun4, gsfc
101    close, lun4
102    free_lun, lun4
103
104
105    ::::::::::::::::::::
106    ::: Translate pixel coordinates in the ctio image to pixel coordinates in the
107    ::: VLT image, and for each object in the latter, compute the distance from
108    ::: this object to every object in the former.
109    ::: This should hopefully help coupling the objects of the two catalogs.
110    ::: For each VLT object, select the object with the smallest distance,
111    ::: and if this is smaller than the tolerance, write it into the final catalog.
112    ::::::::::::::::::::
113
114    ::::::::::::::::::::
115    ::: Open the input and output file, this is cat1
116    ::::::::::::::::::::
117
118    openr, lun3, cat1sorted, /get_lun
119    openr, lun4, cat2sorted, /get_lun
120    openw, lun2, cat1sortradec, /get_lun
121    values = fltarr(7)
122
123    ::::::::::::::::::::
124    ::: Count the number of entries in each catalog, and use these to
125    ::: calculate the number of rows in our intermediate catalog.
126    ::::::::::::::::::::
127
128    nnnarrow = n_elements(vlt(4, *))
129    nnbroad = n_elements(gsfc(4, *))
130    afstarray = fltarr(15, nnbroad)
131    finalarray = fltarr(15, nnnarrow)
132    orphanarray = fltarr(15, nnnarrow)
133
134    print, nnbroad
135    print, nnnarrow
136    row = 0
137
138
139    ::::::::::::::::::::
140    ::: Now, measure all possible distances in the catalogs.
141    ::: This may take some time...
142    ::::::::::::::::::::
143
144    print, 'Calculating distances. This may take a while.'
145
146
147    number = 0
148    for n = 0, nnnarrow-1 do begin
149        xnar = vlt(4, n)

```

```

150         ynar = vlt(5, n)
151
152         x = gsfc(4, *)
153         y = gsfc(5, *)
154         xyad, hdr, x, y, ra, dec
155         adxy, h, ra, dec, xbro, ybro
156         afst = sqrt((xbro-xnar)^2+(ybro-ynar)^2)
157         afstarray(7, *) = afst
158         afstarray(9, *) = xbro-xnar
159         afstarray(10, *) = ybro-ynar
160         afstarray(8, *) = gsfc(0, *)
161         afstarray(11:13, *) = gsfc(1:3, *)
162         afstarray(14, *) = gsfc(6, *)
163         for m = 0, nbroad-1 do begin
164             afstarray(0:6, m) = vlt(0:6, n)
165         endfor
166
167         :::::::
168         ;; Find the entries with the smallest distance,
169         ;; remove the rest
170         :::::::
171
172         best = where(afstarray(7, *) eq min(afstarray(7, *)))
173         afst = afstarray(7, best)
174
175         if afst le offtouse then begin
176             finalarray(*, n) = afstarray(*, best)
177             number = number+1
178         endif else begin
179             orphanarray(*, n) = afstarray(*, best)
180         endelse
181         print, n+1, 'th object of ', nnarrow, ' done'
182     endfor
183
184
185
186
187     finarray = fltarr(15, number)
188     num = where(finalarray(2, *) gt 1.)
189     finarray = finalarray(*, num)
190     print, 'Number of hits: ', n_elements(finarray(0, *))
191
192     Print, "Printing catalog to file"
193     Openw, lun5, "combinedcatalog.dat", /get_lun
194     printf, lun5, finarray, format='(15(f10.5, 2x))'
195     close, lun5
196     free_lun, lun5
197
198     :::::::
199     ;; Now, we find out how many doublets we have, i.e. how many times more than one narrowband
200     ;; object has chosen to identify as the same broadband object.
201     :::::::
202
203
204     print, "Counting doublets"
205     dblcnt = 0
206     doublearray = fltarr(15, number)
207     counter = 0.
208
209     :::::::
210     ;; For any object in the array of coupled objects,
211     ;; check if any of the broadband objects has more than one narrowband object
212     ;; listed as its counterpart. If so, add as many as are found to the counter.
213     :::::::
214
215     for n = 1, number do begin
216         realn = float(n)
217         double = where(finarray(8, *) eq realn)
218         numero = n_elements(double)
219
220         if numero ge 2 then begin
221             dblcnt = dblcnt+numero
222             doublearray(*, counter:counter+numero-1.) = finarray(*, double)
223             counter = counter+numero; ???

```

```

224         endif
225     endif
226     endfor
227
228
229
230
231     :::::
232     ::: Print array to file
233     :::::
234
235
236     doubles = where(doublearray(8, *) ne 0.)
237     numero = n_elements(doubles)
238     darray = fltarr(14, numero)
239     darray = doublearray(*, doubles)
240
241
242     openw, lun7, 'doublearray.dat', /get_lun
243     printf, lun7, doublearray(*, doubles), format='(15(f10.5, 2x))'
244     close, lun7
245     free_lun, lun7
246
247
248     print, 'There are ', dblcnt, ' doubles in this final catalog'
249     print, 'We now look for any systematic offset between the objects'
250
251     ::::::::::::::::::::
252     ::: Now, we plot the x and y direction distance between two incarnations of the same objects
253     ::: to see if there is any systematic error. this is manually corrected by editing the FITS
254     ::: header of the narrowband image
255     ::::::::::::::::::::
256
257
258     deltax = finarray(9, *)
259     deltax = finarray(10, *)
260     plot, deltax, deltax, psym=4
261     oplot,[0,0],[ -1000,1000]
262     oplot,[ -1000,1000],[0,0]
263     nelem = n_elements(finarray(1, *))
264
265     ;           for n = 0, nelem-1 do begin
266     ;               printf, lun5, finarray(*, n), format='(15(f10.5, 2x))'
267     ;           endfor
268
269
270     set_plot, 'ps'
271     device, filename='offsetcontrol666.ps', /encapsulated
272     plot, deltax, deltax, psym=4, xtitle='delta x', ytitle='delta y', title='Offset'
273     oplot,[0,0],[ -1000,1000]
274     oplot,[ -1000,1000],[0,0]
275     device, /close
276     set_plot, 'x'
277     plot, deltax, deltax, psym=4
278
279
280
281
282     ::::::::::::::::::::
283     ::: Now search out those objects that have an excess narrowband luminosity
284     ::::::::::::::::::::
285
286     yval = -2.5*(alog10(finarray(2, *)) - alog10(finarray(12, *)))
287     print, 'finding candidate objects'
288     candidates = where(yval lt -.8)
289     candnum = n_elements(candidates)
290     print, candnum, ' candidates'
291     candarray = fltarr(14, candnum)
292     candarray = finarray(*, candidates)
293
294     openw, lun6, 'candidates.dat', /get_lun
295     print, 'printing candidate array'
296     printf, lun6, candarray, format='(15(f10.5, 2x))'
297     close, lun6
298     free_lun, lun6

```

```

299         print, 'Array of candidates written to file "candidates.dat"'
300
301     ;;;;;;;;;;
302     ;;; Now plot the difference between magnitudes in the two filters.
303     ;;; Mark candidate objects.
304     ;;;;;;;;;;
305
306
307     ;;;;;;
308     ;;; Plot to screen.
309     ;;;;;;
310
311     xval = finarray(0, *)
312     ;yval defined above
313     yval2 = -2.5*(alog10(finarray(2, candidates))-alog10(finarray(12, candidates)))
314     yerr = sqrt((finarray(3, *)/finarray(2, *))^2+(finarray(13, *)/finarray(12, *))^2)
315     zval = finarray(7, *)
316     zval2 = finarray(7, candidates)
317     set_plot, 'x'
318     counter = 1.
319     num = string(counter)
320     plot, zval, yval, psym=1
321
322
323     ;;;;;;
324     ;;; Plot to encapsulated postscript file
325     ;;;;;;
326
327     set_plot, 'ps'
328     device, filename='N-U.eps', /encapsulated
329     device, /color
330     col = getcolor(/load)
331     plot, zval, yval, psym=1, title='U-N vs. virtual distance', ytitle='U-N', xtitle='
    Virtual distance in pixels'
332     errplot, zval, yval-yerr, yval+yerr
333     oplot, zval2, yval2, psym=4, color = col.red
334     oplot,[-1000,1000],[0,0]
335     device, /close
336
337     ;;;;;;;;;;
338     ;;; Now, take care of the objects in the narrow band catalog that have no
339     ;;; counterpart in the other catalog. Are these objects of the desired kind, or...?
340     ;;;;;;;;;;
341
342
343     orphans = where(orphanarray(2, *) gt 1.)
344     orphnum = n_elements(orphans)
345     trueorphanarray = fltarr(15, orphnum)
346     trueorphanarray = orphanarray(*, orphans)
347     openw, lun6, 'orphans.dat', /get_lun
348     printf, lun6, trueorphanarray, format='(15(f10.5, 2x))'
349     close, lun6
350     free_lun, lun6
351
352     print, n_elements(trueorphanarray(0, *)), 'objects in orphanarray'
353     okorphans = where((trueorphanarray(3, *)^(-1))*trueorphanarray(2, *) gt tolerance)
354     print, n_elements(okorphans), 'elements in okorphans'
355     ; print, 'OKOrphans: ', trueorphanarray(*, okorphans)
356     openw, lun6, 'okorphans.dat', /get_lun
357     printf, lun6, trueorphanarray(*, okorphans), format='(15(f10.5, 2x))'
358     close, lun6
359     free_lun, lun6
360     oknum = n_elements(okorphans)
361     okorphanarray = fltarr(15, oknum)
362     okorphanarray = trueorphanarray(*, okorphans)
363     goodorphans = where(okorphanarray(7, *) gt littledist)
364     ; print, n_elements(goodorphans), 'elements in goodorphans'
365
366     if n_elements(goodorphans) gt 1 then begin
367     ;     print, goodorphans
368     ;     print, n_elements(goodorphans)
369     ;     goodnum = n_elements(goodorphans)
370     ;     goodorphanarray = fltarr(15, goodnum)
371     ;     goodorphanarray = okorphanarray(*, goodorphans)
372

```

```

373     openw, lun6, 'goodorphans.dat', /get_lun
374     printf, lun6, goodorphanarray, format='(15(f10.5, 2x))'
375     close, lun6
376     free_lun, lun6
377     print, 'There are ', goodnum, ' entries in the narrowband image with no
        counterpart in the other'
378
379
380     endif
381
382     ::::::::::::::::::::
383     ::: Now, list all elements with a S/N of the above set tolerance or
384     ::: higher in the narrowband image and see how many are also represented
385     ::: in the broadband image.
386     ::::::::::::::::::::
387
388
389     highsns = where((finarray(2, *))/(finarray(3, *)) gt tolerance)
390     numhighsns = n_elements(highsns)
391     Print, 'Out of ', numhighsns, ' ', n_elements(goodorphanarray(0, *)), ' are orphans'
392
393     ::::::::::::::::::::
394     ::: Now plot a histogram of the distances to illustrate the typical distance
395     ::: between incarnations, in order to set a sensible maximum tolerated virtual distance
396     ::::::::::::::::::::
397
398
399     distsort = sort(finarray(7, *))
400     ; print, 'distsort ', distsort
401     distvect = fltarr(n_elements(distsort))
402     distvect(*) = finarray(7, distsort)
403     ; print, 'distvect ', distvect
404     ; print, histogram(distvect, binsize = .2)
405     set_plot, 'x'
406     x = (indgen(n_elements(histogram(distvect, binsize = .2))))/2.
407     plotcolorfill, histogram(distvect, binsize = .2), color = 1
408     set_plot, 'ps'
409     device, filename = 'histogram.eps', /encapsulated
410     device, /color
411     col = getcolor(/load)
412     plotcolorfill, histogram(distvect, binsize = .5), color = col.white;, /midpoint
413     device, /close
414
415     ::::::::::::::::::::
416     ::: Illustrate the relation between the luminosity
417     ::: of an object and the precision of the placing of the center
418     ::: of that object by SExtractor, illustrated in three plots of the distances
419     ::: between the different 'incarnations' of that object.
420     ::::::::::::::::::::
421
422
423     vltpic = readfits('adjusted-HDFScomb.fits', hvlt)
424     ctioptic = readfits('clipped-hdfs_btc_u.fits', hctio)
425
426
427
428     ::::::::::::::::::::
429     ::: First, the faintest objects
430     ::::::::::::::::::::
431
432
433     faints = where(finarray(2, *) lt 60.)
434     print, n_elements(faints)
435     set_plot, 'ps'
436     device, filename='compoffset.eps', /encapsulated
437     device, /color
438     col = getcolor(/load)
439     plot, finarray(9, faints), finarray(10, faints), psym = 2, $
440         title = 'Virtual distances', xtitle = 'delta x/pixels', ytitle = 'delta y/
            pixels'
441     oplot,[0,0],[ -1000,1000]
442     oplot,[ -1000,1000],[0,0]
443
444
445
446

```

```

447
448 ::::::::::::::::::::
449 ::: strong objects next
450 ::::::::::::::::::::
451
452     stra = fltarr(7)
453     stdec = fltarr(7)
454     x = fltarr(7)
455     y = fltarr(7)
456
457     stra(0) = ten(15.*22, 15.*33, 15.*21.288)
458     stra(1) = ten(15.*22, 15.*32, 15.*59.382)
459     stra(2) = ten(15.*22, 15.*33, 15.*05.496)
460     stra(3) = ten(15.*22, 15.*33, 15.*14.085)
461     stra(4) = ten(15.*22, 15.*33, 15.*15.687)
462     stra(5) = ten(15.*22, 15.*32, 15.*39.342)
463     stra(6) = ten(15.*22, 15.*32, 15.*35.931)
464
465     stdec(0) = -ten(60, 30, 59.50)
466     stdec(1) = -ten(60, 31, 19.42)
467     stdec(2) = -ten(60, 35, 56.03)
468     stdec(3) = -ten(60, 32, 12.02)
469     stdec(4) = -ten(60, 32, 24.22)
470     stdec(5) = -ten(60, 31, 22.93)
471     stdec(6) = -ten(60, 30, 56.07)
472
473     adxy, hvlt, stra, stdec, x, y
474
475     sortIndex = sort(finarray(4, *))
476
477     for n = 0, 14 do finarray(n, *) = finarray(n, sortIndex)
478     strongs = fltarr(16, 70)
479
480
481     for n = 0, 6 do begin
482         nearx = where(abs(finarray(4, *)-x(n)) le 30.)
483         neary = where(abs(finarray(5, *) - y(n)) le 30.)
484         num = n_elements(nearx)
485         for m = 0, num-1 do begin
486             closest = where(nearx(m) eq neary(*))
487             if closest ne -1 then begin
488                 strongs(0:14, n) = finarray(*, neary(closest))
489                 strongs(15, n) = n
490                 print, neary(closest), finarray(0:5, neary(closest))
491             endif
492         endfor
493     endfor
494
495
496     goodstrongs = where(strongs(4, *) gt 0.)
497     print, strongs(0:7, goodstrongs)
498     print, x, y
499
500
501     oplot, strongs(9, goodstrongs), strongs(10, goodstrongs), psym = 4, color = col.red
502     oplot,[0,0],[ -1000,1000]
503     oplot,[ -1000,1000],[0,0]
504
505
506 ::::::::::::::::::::
507 ::: intermediate objects next
508 ::::::::::::::::::::
509
510
511     imra = fltarr(7)
512     imdec = fltarr(7)
513     x = fltarr(7)
514     y = fltarr(7)
515
516     imra(0) = ten(15.*22, 15.*32, 15.*44.086)
517     imra(1) = ten(15.*22, 15.*33, 15.*18.585)
518     imra(2) = ten(15.*22, 15.*33, 15.*06.033)
519     imra(3) = ten(15.*22, 15.*33, 15.*15.054)
520     imra(4) = ten(15.*22, 15.*32, 15.*39.026)

```

```

521     imra(5) = ten(15.*22, 15.*32, 15.*44.058)
522     imra(6) = ten(15.*22, 15.*33, 15.*02.691)
523
524     imdec(0) = -ten(60, 34, 57.07)
525     imdec(1) = -ten(60, 34, 38.29)
526     imdec(2) = -ten(60, 33, 50.33)
527     imdec(3) = -ten(60, 31, 13.97)
528     imdec(4) = -ten(60, 35, 36.12)
529     imdec(5) = -ten(60, 34, 57.13)
530     imdec(6) = -ten(60, 32, 13.46)
531
532     adxy, hvlt, imra, imdec, x, y
533
534     sortIndex = sort(finarray(4, *))
535
536     for n = 0, 14 do finarray(n, *) = finarray(n, sortIndex)
537     ims = ftarr(16, 70)
538
539
540     for n = 0, 6 do begin
541         nearx = where(abs(finarray(4, *)-x(n)) le 30.)
542         neary = where(abs(finarray(5, *) - y(n)) le 30.)
543         num = n_elements(nearx)
544         for m = 0, num-1 do begin
545             closest = where(nearx(m) eq neary(*))
546             if closest ne -1 then begin
547                 ims(0:14, n) = finarray(*, neary(closest))
548                 ims(15, n) = n
549                 print, neary(closest), finarray(0:5, neary(closest))
550             endif
551         endfor
552     endfor
553
554     endfor
555
556     goodims = where(ims(4, *) gt 0.)
557     print, ims(0:7, goodims)
558     print, x, y
559
560     oplot, ims(9, goodims), ims(10, goodims), psym = 5, color = col.blue
561     oplot,[0,0],[ -1000,1000]
562     oplot,[ -1000,1000],[0,0]
563
564
565     device, /close
566
567
568
569     print, 'array of candidate orphans written to file "goodorphans.dat"'
570     print, 'Offset plotted, final array written to file.'
571     print, 'Thank you for using COMBINSCATS.PRO!'
572     print, 'we hope you enjoyed the experience and are looking forward to serve you again.'
573
574
575
576
577     close, lun5
578     free_lun, lun5
579
580     close, lun4
581     free_lun, lun4
582
583     close, lun3
584     free_lun, lun3
585
586     close, lun2
587     free_lun, lun2
588
589
590     end

```