Nightview

Filip Hroch

This document describes the Nightview package, an application for control of a CCD astronomical camera along with a telescope mount.

Quick start

- Get and install SBIG library.
- Configure nightviewd server with nightview-conf utility.
- For client, install external pacakges: curl (http parsing library), cfitsio (FITS I/O library) and nightview-shell, gtknightview and telescope, xmove.
- Run nightview and telescoped daemons.
- Test nightview package with nightview-test.

Motivation

Our university was bought a CCD camera a few years before. The CCD acquiring software was enclosed with camera kid but only for Windows or DOS not for linux. So, we was needed a linux software for control of this camera to take a fully advantage of this very expensive instrument.

The developing of this software leaded me to a completely different view to use and control of CCD camera (or some other instrument) with a modern computer. The operation and architecture is a quite un-similar than you know from Windows versions of similar software (ccdoops etc.). I was created a operational schema to use camera in a fully network environment according to the standard unix philosophy. This package has some advantages with respect to others programs. It implements an automatic acquiring of multi-colour image series or co-working with telescope's mount.

My first experiments with the SBIG library leaded me to sbig_exposure, sbig_filter and sbig_temperature shell routines and to a GUI frontend ccdsnaper. The ccdsnaper was a direct pre-release of the GTKnightview. The architecture of this utilities don't supports sharing of resources or any network environment. It's need set suid bit on GTK+ fronted. This (non-secure) feature is not more supported in GTK+ from 1.2 version. All of this leaded my to completely rewriting of these utilities. I was developed ccdsnaper at first quarter of 2001. The switching to a new philosophy was during summer and a new Nightview package was introduced at autumn of 2001. The developing of Nightview was coordinated with developing of xmove (The software for telescope control on *[Monte Boo]* (http://www.physics.muni.cz/mb/)).

NEW! The architecture of the nightview is changed since the verison 0.3.x (february 2002). Now, the server is more secure and more simpler to instalation (the sudo or suid are no-more required).

Brief history

2000, Winter: Linux instaled at MonteBoo
2001, Spring: first experiments with ccdsnaper GUI
2001, July: start of daemon - client framework, renamed to nightview
2001, August: start of test basic functionality of nightview
2001, October: testing completed, debianized
2001, November 15?, nightview published on freshmeat
2002, January: 0.2.x branch frozen, start on 0.3.x
2002, spring: rewrited a network engine, telescope support
2002, June: extensive testing
2002, autumn: packaged, testing finished

Copyright

Feedback

Any feedback is welcome. Please, send any suggestions, notes and bugs to <*hroch@physics.muni.cz*> (mailto:hroch@physics.muni.cz).

WARRANTY

WITHOUT ANY WARRANTY. IF YOU WILL DESTROY ANY EQUIPMENT USING BY NIGHTVIEW PACKAGE IT IS YOU PROBLEM. Be careful, but the Nightview is a relative secure.

Welcome developer

A new developers are welcome. There is driver for a few kinds of the SBIG’s cameras, universal driver for telescope control, but drivers for others instruments missing yet. The CookBook camera, Meade telescope mount (LXxxx) are looking for owners and programers. Please contact me, if you have (or plan write) any another driver.

What is a Nightview?

The nightview is a set of a small simple utilities to a control CCD camera under unix (linux is implemented yet). It is useful for an astronomer to study of the sky by the observation with a CCD camera equipment.

The Nightview package is low, middle and high level package to operate with CCD camera. It uses a multilayer server-client framework, therefore the clients don't depends on specific hardware, only on standard behaviour of these instruments (like acquire of image, select filter and so on). The server depends to a specific HW over its specific library, which operate on communication with CCD camera. Only SBIG camera operation is implemented yet.

The Nightview description

The nightview package is a server-client application. This means that, in principle, minimally two programs must run: The server program on computer with camera connected and some client program. We recommends use different computers for its due to increase efficiency during downloading image over parallel port. Both server and client can run on the same machine, of course.

Our configuration on Monteboo is an old P90 computer as server and (de)Celeron 433 as a platform for client. Both run the Debian. The server don't need massive hardware support. It run only server and a few operational system utilities. I believe that 386 with 8Megs of memory and 50M disc will be OK, but I recommends at least 486 with respect to the higher speeds of image downloading. The more than P90, 16M memory and 100M disc is un-warranted luxury and it don't operate faster because limited by the parallel port and net speed. The server needn't use of disc space. Only the clients saves images. The server-client framework is equipped to use in TCP/IP environment. Than the utility will be work for example over Intranet with Ethernet, serial or radio link. The Monte Boo camera operates on one Ethernet segment of the Masaryk University, so the our camera is directly connected to Internet. Note, that server part not needs some expensive libraries except libsbig, cfitsio and standard system libraries.

The nightview package contains two kinds of clients: GUI for GTK+ and shell utilities. The GTK+ frontend is usable for interactive control of the CCD. It's useful for focusing, temperature regulation or telescope pointing.

The shell client are developed for non-interactive control of CCD. They are contains all functions of the GUI package and all functions of the CCD camera. With this utilities the camera is control over a extremely slow communication lines (telephone) and for the batch processing. There is a shell frontend - night_control - for acquiring of image series with arbitrary combination of the exposure timeouts and filters.

In principle, there is no problem for implementation of clients on other platforms or in others languages or environments. So, the windows users can connect to server with fully transparency as the linux clients for example. Moreover, I suppose that the implementation of the on-line http server with full control of the camera will be developed in the near future.

Building

The Nightview is a relative lightweight application. It needn't the KDE or the GNOME desktops, but it need some libraries.

Note. Please, check this *[ftp://integral.sci.muni.cz/pub/nightview/]* (ftp://integral.sci.muni.cz/pub/nightview/) before building. The Nightview may be compiled here. If you are debian user, please use additional apt source. Add the line to your `/etc/apt/sources.list`: `deb ftp://integral.sci.muni.cz/debian unstable`.

sbig library

This is a basic low-level library for the SBIG cameras by Steve Ashe. You need download it from *Steve's ftp [ftp://ftp.dimensional.com/users/ashe/]* (ftp://ftp.dimensional.com/users/ashe/). Please, read carefully doc enclosed in these archive. Manually install it as root:

```
# tar zxf sbig-linux-x.x.tgz                # x.x is version
# cd sbig-linux-x.x/
# cp sbig.a  /usr/local/lib/libsbig.a
# cp sbig.so /usr/local/lib/libsig.so.x.x    # rewrite version x.x
# cp sbig.h  /usr/local/include/
# cd /usr/local/lib
# ln -s libsbig.so.x.x libsbig.so.x
# ln -s libsbig.so.x.x libsbig.so
```

Now, the libsbig is prepared for use on your system. Please, be sure, that directory /usr/local/lib is in your dynamic's library searching path. The command

```
/sbin/ldconfig -p | grep /usr/local/lib
```

must give an output. If not (usually!), you must add this path to ld cache:

```
# editor /etc/ld.so.config       # add line /usr/local/lib
# ldconfig -v
```

If any application gives error libsbig.so.x.x not found in path..., please check this setting.

Note. I'm preparing the deb package for automatic installation. Please consult actual state on my *[ftp://integral.sci.muni.cz/pub/]* (ftp://integral.sci.muni.cz/pub/) I was prepared a debian package to install this library by dpkg, but this is a installer, you still needs a package's "source". Unfortunately, this instalator is wrong on rpm systems (as Red Heat, Mindrake or Suse).

Second note. The original names of this library are sbig.h, sbig.a and sbig.so. The unix "well-behaved" requires the prefix lib for an every library, so I was renamed the original name.

cFITSIO

cFITSIO[http://heasarc.gsfc.nasa.gov/docs/software/fitsio/fitsio.html] (http://heasarc.gsfc.nasa.gov/docs/software/fitsio/fitsio.html) is a portable library for I/O on an astronomical image format FITS. Clients and server uses this format to save exposure, so you need it. I highly recommend use of the deb or rpm(?) package, which exists at now.

GTK, GDK, GDKpixbuf libraries

I prefer GTK graphics libraries to my development. I know that Qt, Mottif, Athena is better :-), but I prefer GTK. Sorry, if you are uses others. The GTK with GDK support and GDKpixbuf support are a relative small libraries and if you are uses gimp, than you have them. If not, they will be probably in your distribution as packages.

cURL library

This library is used by the clients to the communication over http channel so you need it. cURL is downloable from *cURL[http://curl.haxx.se]* (http://curl.haxx.se) but your distribution files contains probably its.

Compilation

If you have all above libraries installed, you can compile Nightview:

```
# cd nightview
# make [-i]
```

Simply run make. The configure phase is not implemented yet, but it is probably not need because libsbig is strictly depend on PC hardware style. Now, you are waiting for binaries... The parameter -i tells make to ignore errors due to missing utilities or libraries. Use it with caution.

Problems. When the similar message appears in begin of instalation:

```
..
cc -O -D_REENTRANT -I../include -L. -fPIC -c ccdsbig.c
ccdsbig.c:29: sbig.h: No such file or directory
...
```

this means that you don't instaled the sbig.h header file in some system-wide include directory (do you instaled the sbig library?). There is possibility use of the arbitrary directory for this file. Set the shell variable INC_SBIG to this directory. For example:

```
i@hell:~/nightview$ INC_SBIG=/home/my/sbig-linux-2.4 make
```

But this is only instalation help, you still need the install libraries to run the daemon.

Installation

The instalation is need due to simple searching shared libraries (.so) by the binaries. If you get a messages like this:

```
night_power: error while loading shared libraries: libccdnet.so.0:
cannot open shared object file: No such file or directory
```

Your binaries are hungry for instalation. Alternatively you can use setting of the envirnonment variable LD_LIBRARY_PATH (described in Program-Library-HOWTO).

Binaries installation

Installation is normally done by typing (I'm not sure..)

```
# make install DESTDIR=/usr/local
```

It will be install packages under /usr/local directory (to bin and lib).

Documentation

The documentation is an important part of the package. It was written in the DocBook XML. So, you need installed a xml pre-processor, jade and jadetex for ps/html, pdfjadetex for pdf and xsltproc for html version of the document. Alternatively, you can use Mozilla or Opera to view this document directly as a simply structured xml. Please, ignore errors due to missing jade, jadetex, pdfjadetex, xsltproc or DocBook during compilation (the switch -i can be useful). The Nightview will be correctly work without any documentation.

You can tune some parameters for documentation. The variable JADEHTML in doc/Makefile control the html generation by jade (set to 1) or XSL preprocesor (set to 0). Try and use nicer alternative for you. Please, set additional variables (XSL and DSSSL stylesheets) to your system dependend values, if you have another directories system than I have.

The full documentation is also available from Nightview download site.

Configuration

nightviewd

The main nightview server is the nightviewd. I reccomends start this server at a boot time from any rc.d file. Many systems have directory /etc/init.d/ or /etc/rc.d/ where the start scripts appears. The debian users can use the prepackaged start script. If you can start nightviewd at boot time, edit some start script (/etc/init.d/local for example) and put this line to the file:

```
/usr/sbin/nightviewd
```

Command line parameters for nightviewd:

```
 -device 0x378 (default) | 0x3bc |  0x278
        lp0             | lp1   | lp2    ... IO address of the paralell port
 -filter1 x  -filter2 x ... filter5 x
        define filter specification up to 5 filters
 -site Site Name
 -longitude Longitude of the site
 -latitude  Latitude of the site
 -altitude  Altitude of teh site
 -telescope Telescope designation
```

All above parameters are specified in the /etc/nightview.conf file. It's generated by nightview-conf script. This file is readed first time. The command line parameters replaces the config file values.

Note. The nightviewd is not a well-matured daemon for now. It can crash everytime. I highly reccomends (if you haven't root access) use some from utilities to daemon monitoring and restart it during observation. The daemon create file (socket) /tmp/.night_shock.

The nightview package contains a script nightview-conf to create the configuration file for the nightview server. The debian instalation packages will be created automaticaly, but you can run it by the hand. The generated file has defaults used by the daemon when no configure and no command options appears. The created file /etc/nightviewd.conf follows:

```
# Config file for Nightview daemon, server part.
#
#

# ----------------------------------------------------------------------------
# HW address port specification
#
# Select one from this:
#
#   port      device      address
#    1        /dev/lp0    0x378      (default)
#    2        /dev/lp1    0x3bc
#    3        /dev/lp2    0x278

Device HW address = 0x378

#
#-----------------------------------------------------------------------------

#----------------------------------------------------------------------------
# Debug options
#
# Select level:
#
#   level      means
#    0         no debug informations, only status codes
#    1         print errors to standard error
#    2 - 5     from 2 to 5 increase print of debug informations
#
#    Use 0 for normal operation, increase when you have a problem.

Device HW debug level = 0

#
#----------------------------------------------------------------------------
# Site options
#
```

```
# Specify your site coordinates. This values will be used without any
# changes in header of the created FITS file.
#

# Name of site
Site = "Monte Boo"

# longitude (+east, -west) in degrees
Longitude = 16.58395

# latitude (+north, -south) in degrees
Latitude = 49.204128

# altitude (+over sea, -under sea) in meters
Altitude = 304.0

# telescope
Telescope = "0.62 m, 1:1.44 refl."

#
#---------------------------------------------------------------------------
# Filters definition
#
#  Specify string(s) for your filter(s)
#

# position in carousel = filter
Filter 1 = "B"
Filter 2 = "V"
Filter 3 = "R"
Filter 4 = "I"
Filter 5 = "clear"

#
#---------------------------------------------------------------------------
#
```

There is a possibility to change address of the connected parallel port. The switch `Device HW address = 0x378` for a first port (default) or `Device HW address = 0x3bc`for a second port in the /etc/nightview.conf file. This file provide the run-time debug level configuration also. Run `nightview-conf` script to create it. See these file for more description.

Generally, If you have the camera connected to the first parallel port, the HW address will be correct, eg. no changes are need for first tests. The new computers have two parallel ports on board, but extremely rarely are both connected to connectors.

The other parameters are used for fine tunning. The debug option is for hackers only (see description of debug option in sbig library). The site parameters idenfify your observation station. The values are no direcly used by the nightview package. They’re copied to the FITS header without any change.

The filter options defines your filters. The values are any strings enclosed to the apostrophes (`"`). The night_filter -list command will be print this definition in clients programs.

The running kernel needs to contain the paralell port support. Modules are preferred. Set these variables:

```
CONFIG_PARPORT=m
CONFIG_PARPORT_PC=m
CONFIG_PARPORT_PC_CML1=m
```

(this function did’t fully tested)

The server has changed logging facility from the 0.3.0 version. The syslog is used now. Any critical events (server launch, client connect, errors) will be included in syslog file. All others list from daemon are debug messages to stderr. They are enables the define the DEBUG macro in ccd.h. This loging will be disabled per default in future (in stable version).

Tip. If you have a server’s computer on the net I reccomends use ntpdate and ntpd to synchronise time with some time server. Include to your `/etc/crontab` this line:

```
58 * * * * root ntpdate -u 999.999.999.999 >& /dev/null &&  hwclock --systohc
```

The ntpdate will be synchronize your computer once per hour with the specified timeserver. The list of time servers can be found at page: ...

http server

The nightview needs a http server with CGI support to run over internet (for local clients only is not need). You can use any http server but I strongly recomends a small and a fast server like thttpd *thttpd by Jef Poskanzer [http://www.acme.com/software/thttpd/]* (http://www.acme.com/software/thttpd/) or boa. *boa by x x [http://www.boa.org]* (http://www.boa.org) Use it in standalone mode (run as own procces not lunched from the inetd, moreover both boa or thttpd don’t support the inetd, but wn *wn by x x [http://www.wn.org]* (http://www.wn.org) for example supports it) for a shorter response.

The scripts nightview-thttpd and nightview-boa runs the http server as system user.

I use the thttpd server. There is a script can be used for a lunch this server in a chroot envirnoment in tmp directory, where the server create a log file and some temporary files.

```
# mkdir /tmp/thttpd
# ln -s /usr/bin/nightview.cgi /tmp/thttpd
# thttpd -d /tmp/thttpd -p 7666 -r -c *.cgi -l /tmp/thttpd/thttpd.log
```

The thttpd server listen on 7666 port (you can use 80 also, but the another non-nightview http server can run on this port, if you can use #80 use -host "name:80" option for clients). The all network traffic on this server will be logged to the thttp.log file.

To testing of the server use usual web browser. Copy the nightview.html file to the /tmp/thttpd directory and look at:

```
lynx http://localhost:7666/nightview.html
```

after a little bit of time appears the page:

```
    This is a simple tester for nightview web interface.
    If you don’t know how use it, type "login" to some entry.
    TELESCOPE: ________________________________________
    CAMERA: ________________________________________
    Click
    ...
```

Type "login" and the server send you a some message. My lynx runs the mozilla when I send the login message to the server due to sending XML back.

Testing of instalation

The basic test is an output from ps. It must show the nightviewd:

```
debian:~/nightview/shell$ ps axwwu | grep nightview
USER       PID %CPU %MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
...
root       664  0.0  0.7  2856  920 pts/1    S    19:05   0:08 ./nightviewd
f         1774  0.0  0.4  1556  540 pts/2    S    23:53   0:00 grep nightview
...
```

Ok. Nightview server runs. Than we can run the nightview-test script. It was developed to the simple tests of basic functionality of the nightview daemon and clients. Switch camera on and run it. The complete local environment (server and clients run on the same computer) will be tested. The http server can be down during this test.

If your camera works locally, you can try connect over internet. Use -host option in command line.

User guide to Nightview

The GTK and shell interface to the nightview are implemented yet. Anybody can add other clients.

Shell clients

There are a ’night_exposure’, ’night_temperature’, ’night_filter’, ’night_power’ and ’night_control’ shell utilities to control exposure, temperature, filter and batch use of a CCD camera. The night_control is only a wrapper to get series of an astronomical images with different exposure times and filters.

All of utilities have standard behaviour. They are driven by the command line parameters. It can be useful for scripts or single command from shell. The common options of all commands are an internet address of the server and its port. Run its without parameters to get short description of its options.

night_exposure

It is invoked with

```
night_exposure [-t time] [-s shutter on|off] [-b 1|2|3] [-o output]
[-r ’(x1,y1,x2,y2)’] [-o ’filename’] [-name ’objectname’] [-host address]
```

where -t time is exposure time in seconds (real number). -o output gives name of the output file (default: nightview.fits). The binning can be -b 1,2,3 meaning 1x1 (full resolution), 2x2 (half resolution) and 3x3 (every pixel of image is composition of 9 chip's pixels). This switch don't change the area for readout. The readout area can be changed with -r option. The -s switch control the shutter (useful for dark frames). You can specify an object name for use in 'OBJECT' keyword of the FITS file header (defaulted to null). The common options -host is defaulted to file:///tmp/.night_shock set the comunication socket to a CCD camera.

The region is selected by four integer numbers enclosed in braces. The coordinates starts at left bottom corner according to the standard mathematical convention (eg. not stupid computer graphics convention!). The (1,1,100,100) selects a left bottom part of image. Don't forget use of apostrofes, your shell minds braces differently than you.

The switch -r (region) is useful for a faster downloading but only of some part of a image. For example, it can be used for acquiring of the series of the exposures with a fine time resolution (measuring of periodic error of mount or observation of occultation?).

The options after -o and -name wonts a string. If you use a multi word name or an exotic character (please, suppose that all other a-z, A-Z and 0-9 are an exotic for your shell) enclose this string to apostrophes '.

night_filter

Nigh filter controls a color wheel. It's developed to set of selected filter and list of possible filters. It is invoked with

```
night_filter [-f filter] [-list]
```

The basic philosophy is list the possible filters with -list option. Than you can select any filter from this set by the use -f option. The list is generated by nightview server so the superuser can define the names of filters in the server's configure file. See above. The common options -host is defaulted to file:///tmp/.night_shock set the comunication socket to a CCD camera.

night_power

Nigh power controls a basic function of camera. It switch up, down of the power and print basic info. It is invoked with

```
night_power {on|off|info} [-host address]
```

Without parameters prints help. The 'on' connect to camera and switch it on. The 'off' option switch camera to down. The 'info' option prints basic camera info (firmware description, number of pixels, pixel's sizes and so on) if camera is on. The common options -host is defaulted to file:///tmp/.night_shock set the comunication socket to a CCD camera.

night_temperature

It is invoked with

```
night_temperature [get|set] [-t|-ta temperature] [-r] [-off]
```

The basic modes of this command are an informational ’get’ (print ccd, air temperature, fan status and cooling power) and ’set’ (set the ccd temperature to specified temperature or set off temperature regulation). The common options -host is defaulted to file:///tmp/.night_shock set the comunication socket to a CCD camera. Note, it is advised to turn off regulation, but leave the fan on for a minute or so prior to shutting down the camera.

night_control

It is invoked with

```
night_control [options]
```

where options are:

```
-e exposure time(s) in seconds, comma separated values of real numbers for
   specified filters
-f filter, specify one or more from xBVRI, default x
-o output filename mask, default nightview.fits, mask in C format rules,
   integer first, character last
-n number of exposures, default=1
-d dark interval, default: 0 no darks
-b binning, default=3 (directly passed to night_exposure)
-r select region by format: ’(x1,y1,x2,y2)’, default: full area
-name object’s name (directly passed to night_exposure)
-t test, only prints commands, useful for background processing
-v view image after every exposure
-h print help
```

Note for non-C programers. The format %d will be print integers without leading zeros. The format %02d will be print integers with two decimal places: 01,02,03,04..09,10,11..99. Analogically for %03d and so on. The format %s will be print any string.

Example1: The astronomer wonts 100 exposures of PR Del, each 15 second in V filter, low resolution:

```
night_control -e 15 -n 100 -f V -name ’PR Del’ -o ’prdel%03d%s.fits’
```

Example1: The astronomers wonts 100 exposures of PR Del, each 15 seconds on V filter, low-resolution, every 20th exposure dark:

```
night_control -e 15 -n 100 -f V -d 20 -name ’PR Del’ -o ’prdel%03d%s.fits’ -v
```

Example3: The astronomers wonts 100 exposures of PR Del, each 15 seconds in V and 10 seconds in R filter, low-resolution, every 20th exposure dark:

```
night_control -e 15,10 -n 100 -f VR -d 20 -name ’PR Del’ -o ’prdel%03d%s.fits’ -v
```

Technical note 1. The night_control search the binaries in standard system directories by default. So, if you need a run they from some non-standard place add this directory to the system variable PATH.

Environmental variables

All shell utilities (night_*) gets information from enviromental variables. The possibility of specification the ’-host’ option is implemented at now. The comamnd line options replaces the enviromental setting in general.

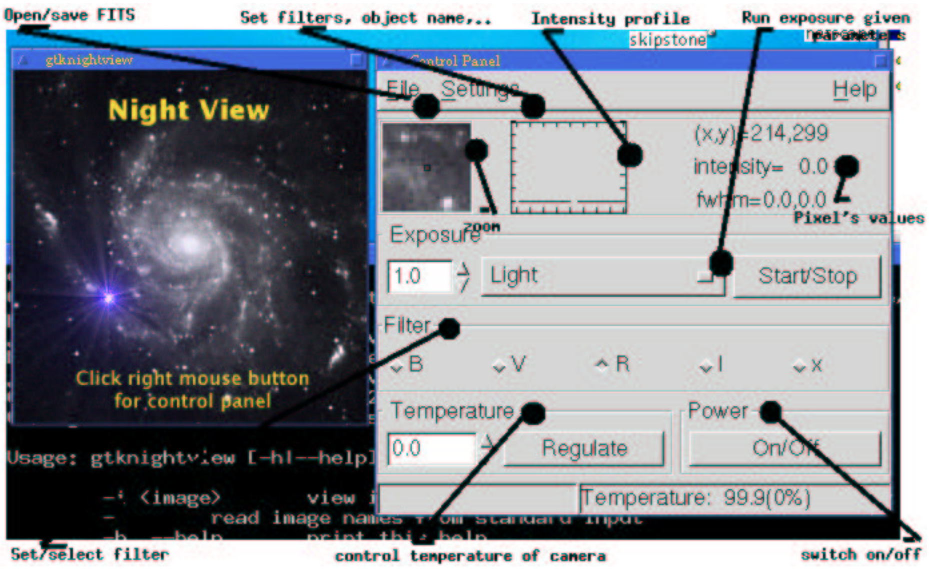
NIGHTVIEW_HOST variable sets the ’host’ option commonly for all shell utilities. If varieble is unset this variable is defaulted to file:///tmp/.night_shock (local connection).

GUI interface

The GUI interface is named ’gtknightview’. You can control a CCD camera with it and view some FITS files. The interface is intuitive. It can be invoked with

```
gtknightview [-h|--help] [-i image] [-]
```

some optional parameters. The user can ask for help. The user can specify an image to view. The FITS images will read from standard input with "-" specified on command line. In this case, the nightview will be closed after end of file. This is mode for batch exposing. The nightview will be used together with night_control.



Main window and control panel of nightview. The basic description is included.

The gtknightview is designed to an iterative operation with CCD instrument, not for a huge series of your CCD images. Please, use the night_control utility for this purposes. I don’t plan integration any series-like feature to the gtknightview’s menu. It’s possible that another graphics utility will be written as a frontend to night_control.

Note. It’s recommended use of the lbxproxy or the dxps utilities for communication with server over a slow link.

Setting output files as read-only

In principle, every image from the CCD astronomical camera is unique. No repeating of heaven's mysteries is possible. Therefore, the utilities of nightview package creates every saved image as read-only by default. (Technically the umask mode is set to 444 on Unix systems.) There is no switch for overwriting of this feature except patching and re-compilation of sources. If you can rewrite or delete your data (NOT RECOMMENDED) use chmod command for this purpose.

I was added this feature after unfortunately removing all of my images of the 1998WT24 asteroid, BL Lac observations and time series of SAX xxxx+xxx. Please, be careful.

Mount of telescope

The nightview package also provide the server for telescope's mount together with shell a GTK+ client. They are control the telescope motions, clock and a dome slit position.

Mount's server

The server is a simpler variant of the nightview server. It listen on socket, it knows a set of commands. The server is a multi-threaded. A first thread communicates with clients, a second controls a timers, start mount's engines and updates time specific variables (Julian date, azimuth of the telescope.. or spin the dome is it is need) and a third periodically saves the coordinates to file (for more robust behaviour when something crashed). The observation site parameters shares with nightview package. The server is a relative clever but it isn't the oracle. It uses all coordinates in degrees but it don't know nothing about your steeper motors. Therefore, the user should provide a few a low-level utilities (described in detail below) to do it.

Config file

The mount config file is shared with nightview daemon. It's means this file is /etc/nightview.conf. Only those items are used by the telescoped:

```
Longitude = 16.58395
Latitude = 49.204128
```

The geographical coordinates of your site in degrees must be provided. The longitude is coded as positive on east and negative to west with respect to zero's meridian (eg Greenwich). The latitude is in obvious manner. This values are used to computation of horizontal coordinates, Julian time etc. so enter the most precise values you knows.

Invoking options

Both latitude and longitude parameters can by independently set on command line during invoking of the daemon. The options are: -latitude and -longitude. See this example:

```
telescoped -latitude 49.204128 -longitude 16.58395
```

The command line switches overrides the config file setting. For details see the config file description.

Note. If command line switches are not specified and config file is not found, the defaults (Longitude = 16.58395, Latitude = 49.204128) are used. They are unusable for you I suppose.

Note 2. Any part of this package wasn't tested on a wild west or a heat south. Be careful.

Specification of the mount drivers

The server is independent on your hardware solution of the mount. This independence requires a site specific solution. This solution is build on a few low-level subroutines directly used to communication with mount engine.

A good example is a mount with axes driven by steeper motors. The steeper motor have 100 steps per revolution (for example), your gears provides 1:3600 ratio so the low-level subroutines may know that you have a 100 steps per degree. Moreover, this drivers may provide communication over some interface (hardware dependent again).

The nightview package defines six general commands to control of the telescope. The `telescope_rekl`, `telescope_dekl`, `telescope_clock` and `telescope_stop` controls the telescope. The `telescope_domeinit` and `telescope_dome` controls the dome. The table describe usage:

Table 1. Low-level commands used by telescoped

command	options
telescope_rekl	difference [deg] velocity [deg/dec]
telescope_dekl	difference [deg] velocity [deg/dec]
telescope_clock	0 | 1
telescope_stop	
telescope_domeinit	0 | 1
telescope_dome	difference [deg] velocity [deg/dec]

The `telescope_rekl` command change the Right Ascension of the telescope about value difference. If actual value of the telescope is and the difference than the telescope is moved to:

Equation 1.

The difference are in degrees, positive and negative values. Second parameter is velocity of moving in degrees per second. The command should immediately print the number of seconds needs to moving as a float number. If any error is occurred the zero is printed and return value should by set to non-zero.

The `telescope_dekl` command change the Declination of the telescope about value difference. If actual value of the telescope is and the difference than the telescope is moved to:

Equation 2.

The difference are in degrees, positive and negative values. Second parameter is velocity of moving in degrees per second. The command should immediately print the number of seconds needs to moving as a float number. If any error is occurred the zero is printed and return value should by set to non-zero.

The `telescope_clock` command switch on the clock of the mount when parameter 1 (or any positive non-zero value) is supplied. If the zero (or negative or nothing) is specified, the clock is stopped. The float-number format can be used. The command should immediately print the number one (1) (may by a float number) when started and number zero (0) (may by a float number) when stopped the clocks. If any error is occurred the zero is printed and return value should by set to non-zero.

The `telescope_stop` command immediately stops all moving (clock included) motors. It’s normally not used and is invoked when unconditional interrupt from user is occurred. No return or print values are checked or used.

The `telescope_domeinit` command initialise the dome (open the doors, spin to the telescope position for example) the 1 when parameter 1 (or any positive non-zero value) is supplied. If the zero (or negative or nothing) is specified, the dome is uninitialised (doors are closed, power is down). The float-number format can be used. The command should immediately print the number one (1) (may by a float number) when dome is initia and number zero (0) (may by a float number) when uninitialised the dome. If any error is occurred the zero is printed and return value should by set to non-zero.

The `telescope_dome` command change the azimuth of the dome’s slit about value difference. If actual value of the slit is and the difference than the dome is spin around angle:

Equation 3.

The difference are in degrees, positive and negative values. Second parameter is velocity of moving in degrees per second. The command should immediately print the number of seconds needs to moving as a float number. If any error is occurred the zero is printed and return value should by set to non-zero.

Mount’s clients

The shell and GTK+ clients are provided by the nightview package. The shell client is named `telescope` and GTK+ client is `xmove`.

telescope

The main utility to control of the telescope is `telescope`. It’s command line driven. No any config files are supposed. Telescope provide setting of getting many important characteristics of the dome, telescope or date. Transparently adds support for the local or Internet connection. The on-line help is provided with invoking it without parameters:

```
Telescope mount position control 0.0.0
Use: telescope [set | get] [options, values]

         get:    (specify nothing, one or more option)
                 -ra   print Right Ascension in degrees
```

```
                -dec  print Declination in degrees
                -a    print Azimuth (180 deg = north)
                -z    print Altitude
                -ha   print Hour Angle
                -jd   print Julian date
                -status   print status of moving telescope

        set:    (every option needs additional parameter(s))
                -coo RA Dec  set the Equatorial coordinates in degrees
                -cal RA Dec calibrate the coordinates (degrees)
                +ra   add value to Right Ascension in degrees
                +dec  add value to Declination in degrees
                -vra   set velocity of Right Ascension in degrees per second
                -vdec  set velocity of Declination in degrees per seconds
                -c on   start clock-machine
                -c off  stop clock-machine
                -c park          park telescope

                -host address:port Internet address of server, "localhost:7666"
                -q don't print the telescope status during setting of telescope
```

Two main modes are supported. The get mode for getting all or specified characteristics and set mode to set and calibrate values. The key set or get should be used to identify mode. The additional parameters can follow.

The get options switch to get mode. Than user can specify one, more or nothing option and required values will be print. The format of the output list is list of items, each

```
name = value
```

where name is a name (can include spaces) of the characteristics and value as a float number. If any parameter from set (jd, ra, dec...) follows the parameter as printed without leading name and a equal sign. Only float is printed.

When you specify the get option on the command line without others parameters all of characteristics will be printed:

```
debian:~/nightview/mount$ telescope get
Right Ascension =   0.000
Declination     =   0.000
Azimuth          = 156.238
Altitude        = -37.523
Hour angle      =  17.363
Dome            = 111.693
Julian date     = 2452372.3623
Siderical time  =  10.000
```

The one (or more) parameters can be added:

```
debian:~/nightview/mount$ telescope get -jd
2452372.364
```

The setting of various coordinates is more exciting. The coordinates are calibrated by using the -cal (abbreviation of calibrate) options follows by two float numbers meaning the current coordinates. This command print 1 on success and 0 in failure and return code is set. For example, you cantered telescope to some bright star by hand and you need tell to server when the coordinates of this star are the current. This options will be frequently used at start of observation.

```
debian:~/nightview/mount$ telescope set -cal 10 10
1
```

When the coordinates were calibrated, we can point the telescope to some interesting object with -coo option (abbreviation of coordinate) followed by the two float numbers as required coordinates again. This command print 1 on success and 0 in failure and return code is set. The motors will be switched on and telescope will be moving. The utility prints text pseudo-progress bars when telescope is moving, with switch -q this behaviour is off-ed and this routine exits immediately.

```
debian:~/nightview/mount$ telescope set -coo 11 11
1
```

The moving velocity in both axes should be separately set before any moving is executed with parameters -vra (velocity RA) and -vdec (velocity dec) followed by float numbers. This velocity is directly passed to low-level subroutines already described. This parameters should be used together with -coo option. When velocity isn’t specified is set to 1 degrees per second.

```
debian:~/nightview/mount$ telescope set -vra 1 -vdec -coo 11 1
1
```

The relative difference can be specified too. Then use switches +ra and +dec (+ mean addition to current coordinates). The differences in degrees will be add to current coordinates. The moving velocity can be specified, otherwise is set to 1 deg per second. The utility prints text pseudo-progress bars when telescope is moving, with switch -q this behaviour is off-ed and this routine exits immediately.

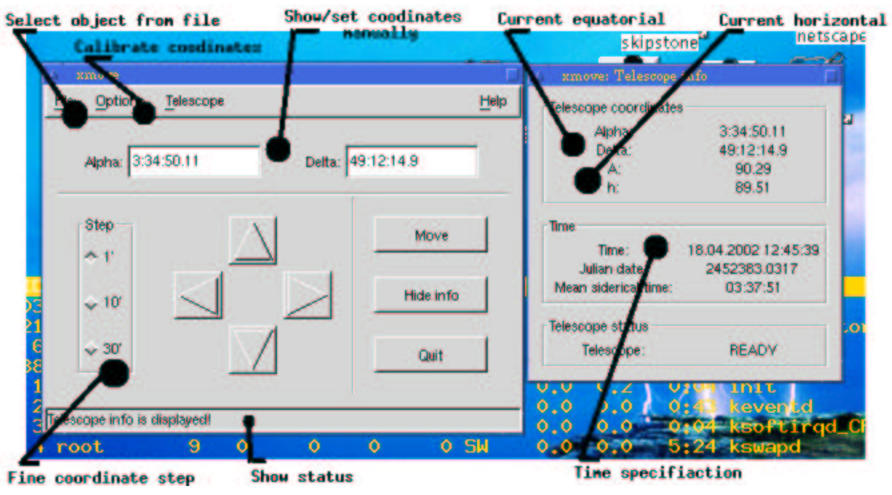
The switch -c followed value from a set (on, off, park) controls the clock. The option on switch up the clock, off stops the clock and park move telescope to park position (azimuth 0, height 90). This command print 1 on success and 0 in failure and return code is set.

```
debian:~/nightview/mount$ telescope set -c on
debian:~/nightview/mount$ telescope set -c off
```

This utility connect the local socket when the option -host is not used. If you can connect to a remote server than you should use the -host option followed by the Internet address of the server.

xmove

...description of this utilities..



Alpha: 3:34:50.11 Delta: 49:12:14.9

Telescope info is displayed!

Telescope coordinates

Alpha: 3:34:50.11
Delta: 49:12:14.9
A: 90.29
h: 89.51

Time

Time: 18.04.2002 12:45:39
Julian date: 2452383.0317
Mean siderical time: 03:37:51

Telescope status

Telescope: READY

Main window and control panel of xmove. The basic description is included.

FAQ

1. Hardware specific questions

Q: Do operates nightview with ST6 or older cameras connected over a serial port?

A: No. The low-level libsbig library don't supports this feature.

To Do

Correct my English.

Test, test, test... I'm planing only fixing of bugs at near future.

Test client side on non-Linux systems.

Native packaging for RPM based systems

Test the tracking chip.

There is a strong need for web interface. The Java, PHP programers and designers are welcome.

Bugs

The server and clients don't supports authentification. Anybody can connect your camera when the server is running. Please, run a simple firewall on machine with camera connected.

Changelog

The versions up to 0.2.3 snapshots original nightview architecture with server listening on network socket.

The version 0.3 and after implements a secure model with communication over http protocol.

Programer’s guide to the nightview package

You can use any part of the code to your application so I will try describe some interesting moments in this package.

The communication between clients and server has two parts. The local, when the clients directly send data to the server socket. This is the same mechanism as pipe "|" in shell for example. The local socket is a special file (ls -laF shows /tmp/.nigh_shock=) like a named pipe. The client send a text strint to this socket, server read it from this pipe and send back a response.

The server run as root (needs by sbig library) so the direct connection to the outher world is a relative dangerous. Therefore, the server listens only local connection over the local socket. The use internet connection is wrapped over the http protocol by usual mechanism cgi-bin questions and ansfers. The data for clients are enapsulates to the simple XML file for more effective handling.

The ccdnet library is developed as a mechanism for obscuring of the local or non local connections for the user. This connection library has an own object (structure) for a connection with parameters for local or non-local connections. This mechanism simply obscures the communication mechanism and user lives under virtual reality that the communiaction is the same.

The com.h is a main definition part of this connection library. It’s containing the definition of the TFILE structure:

```
struct _TFILE {
  int id;                  /* low-level identificator of chanell */
  int local;               /* = 1 for local connection (unix socket) , else 0 */
  char *key;               /* user defined string as ’key=value’ in cgi and <key>
  char *host;              /* URL of server */
  int http_size;           /* size of private http buffer in bytes */
  char *http_buffer;       /* pointer to the private buffer */
};
```

The behavior of all public functions (methods) depends on the shape of the client address. It’s based on widely used URI convention. If the address starts with string "file://" than the connection is a local and the local connection is initialized (the TFILE.local is set to 1, and TFILE.id contains file descriptor by of some soket). Opposite with this, if address starts with "http://" the connection is non-local and network connection is initialized. The TFILE.host points to the address and the TFILE.tag points to "camera" strig when we initialized connection to the camera and "telescope" for connect with telescope driver. This is initialized by the

```
TFILE *topen(char *host, char *key)
```

subroutine. The others functions has an analogical behavior as the stdio functions fread, fwrite and fclose:

```
int tread(TFILE *t, char *buf, int size)
int twrite(TFILE *t, char *buf, int size)
int tclose(TFILE *t)
```

The com.c file provides also the file_download function used to download the file over network (simply send the request like http://localhost:7666/file.fits and saves this file.

The network http connection is usual cgi-bin POST request, the server sends data to the client enapsulates in XML document. The httpsub.c file defines the function http_run and http_download used to work with cURL library to sending cgi request and provides response (to TFILE.http_size and TFILE.http_buffer) or download remote file.

The XML document generated by remote server has a simple structure. It is described to the simple parse of the response. The file smlsub.c search this file for a TFILE.tag and gets the string between start and end tags. The DTD doesn't exist so not the validation of this file isn't possible. There is a typical structure:

```
<?xml version="1.0"?>
<status>
<camera>welcome par1 par2</camera>
</status>
```

The implemented tags are camera, telescope and error. When error tag occures the problem is in the cgi script because error is generated only by the cgi routine not by a daemon.

The client and server comunicates over a set of string (commands). The protocol string was been created for all elemnetary operations with camera. This strings (like regular shell commands) provides commands and passing arguments of them.

The common compiler of the C is a c89. c89 is a standardised command for all Unixes with equal set of options (like f77 or f90). The problem is that this command doesn't recognise useful functions as strdup or snprint. The C is highly non-standardised language.

I'm using only the C and perl program languages. The fortran (ADA too) is not useful when it's a high-level language and don't support HW depended sockets (for example). The use of other scripting languages like python was due to its abnormally high system requirements canceled.

ccd library

The information about conected camera is included in CCD structure.

```
typedef struct _CCD CCD;

struct _CCD {
  int device_address;              /* HW addres of device */
  char *address;                   /* general type of address */
  char id;                         /* identificator (file,net) */
};
```

A. Version of this document

$Id: nightview.xml,v 1.21 2002/10/30 17:34:23 hroch Exp $