LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
-LIGO-
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note	**LIGO-T030283-00-C**	12/03/03

ROBO PLOTTER

Chethan Parameswariah

This is an internal working note
of the LIGO Project.

LIGO Livingston Observatory
19100 Ligo Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352-0159
Phone (509) 372-8106
Fax (509) 372-8137

California Institute of Technology
LIGO Project – MS 51-33
Pasadena CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project – MS 20B-145
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

WWW: http://www.ligo.caltech.edu

1. ABSTRACT

This document describes the working of the automatic plotting and web logging system called "ROBO PLOTTER" for plotting 'Daily Summary Plots' and logging them into the LLO elog at 16:00 hours UTC every morning during science and engineering runs. Robo-plotter is part of the collection of software robots - "*SOFT-ROBOTS*" now working at LLO to ease and improve efficiency.

2. INTRODUCTION

Automation of LIGO CDS Systems is essential to maintain consistency and to minimize human errors, with the advancement of interferometer into low noise commissioning and science runs.

ROBO PLOTTER is a software robot that collects minute trend data from the important channels defined by the configuration file for previous 24 hours and then plots it as a postscript file. A pdf of the postscript file is also created and then elogged to current day's elog at 16:00 hrs UTC during the science runs.

3. OPERATION

ROBO PLOTTER is part of "*SOFT-ROBOTS*" - a collection of software programs and scripts called '*software robots'* that are run at LIGO to automate various duties of the scimons, operators and engineers to ease and improve efficiency while maintaining reliability and consistency on a daily basis.

The main program "robo_plotter.pl" is a data plotter and auto e-logger program. This is a Perl program that is spawned by a cronjob shell script "robo_plotter.sh" every morning at 10.00 AM (11:00 AM – during daylight saving time) and collects the minute trend data for the previous 24 hours from 16:00 UTC today to 16:00 UTC yesterday and e-logs the collected data on the today's page of the e-log. Currently this cronjob is run on control7 in the control room at LLO and the program requires this sun workstation to be 'ON' for it to work.

The program listing for both robo_plotter.sh and robo_plotter.pl (version 1.0) is in Appendix.

The shell script robo_plotter.sh sets the shell and the display environment to correct values before spawning the perl program robo_plotter.pl.

The perl program robo_plotter.pl first has a set of variables hard coded into the program such as the location of required perl modules, science run start date, IFO identifier, and then calculates the current and yesterday's gps times.

With this information, a command line dataviewer (written by Hongyu Ding at Caltech) which produces a postscript file as the output is spawned with the right arguments such as the server ip, port number, configuration file, start time, duration, conversion flag and the output file name.

The configuration file called "S3DailySummary.xml" file has a certain format and is also listed in Appendix. The details of the format are in the "README" file listed in Appendix.

The output file is a postscript file with a correct file name (with yesterday's month and date) that is automatically generated by the robo_plotter.pl perl program. This output file is saved in the right location and then converted to a pdf document using the "ps2pdfL" command line distiller. The perl program spawns the distiller after a delay of 10 seconds to ensure the output file is generated first. The pdf is also saved along with the postscript file with the same name but with the '.pdf' extension.

The perl program then submits to elog, a form with the pdf file, to be posted on today's LLO elog.

Figure 1 shows the posting of the "Daily Summary Plot" on the LLO elog done during science run S3.

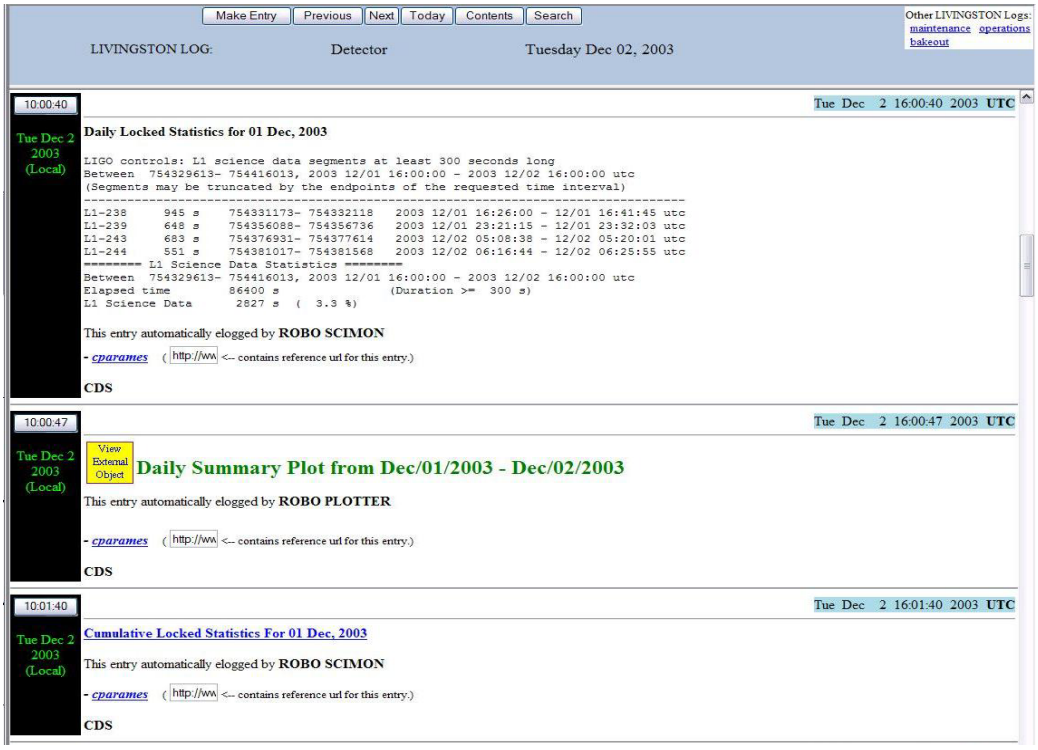

Figure 1: Daily Summary Plot entry automatically posted elogged by ROBO PLOTTER.

Clicking on the “View External Object” link (yellow box in the entry) opens up the “Daily Summary Plot” in pdf format. Figure 2 shows a daily summary plot for a one day during S3.

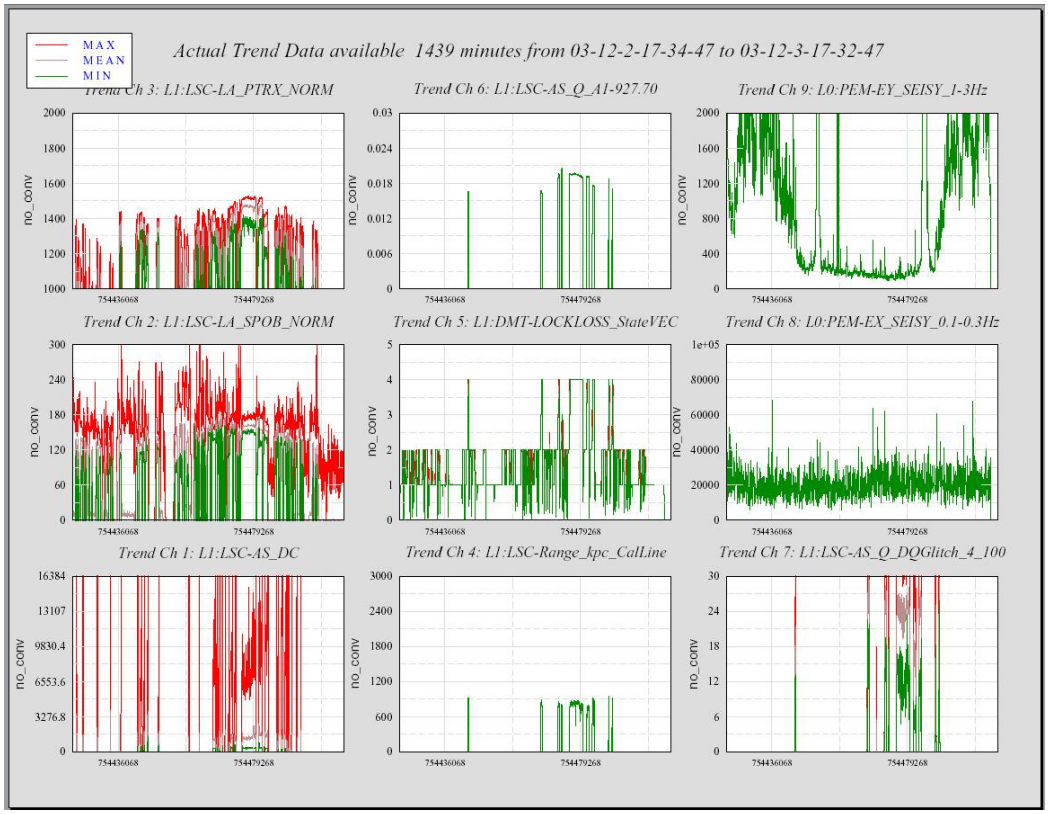

Figure 2: Daily Summary Plot

4. CONCLUSION

The ROBO PLOTTER was initiated for the science run S3 and has been working since. A few bugs in the command line dataviewer with regards to auto-scaling and conversion were fixed. Features such as operator start and stop, day light savings time identification, and the science run start and stop date information is currently either non-existant or is hard-coded in the program. The next version of the Robo-plotter will automate this adding intelligence to the program.

APPENDIX

Note: All the programs and configuration file are located on LLO CDS machines in the directory - */cvs/cds/project/roboplot/*. The output postscript and pdf files are located in the directory - */opt/LLO/c/ops/public_html/S3/DailyStatistics/*.

1) Shell script run as cronjob at 10:00 AM (11:00 AM during daylight saving time) - robo_plotter.sh

```
#!/bin/csh
setenv DISPLAY control7:0.0
setenv DVPATH /cvs/cds/llo/target/sun/G2.2
/cvs/cds/project/roboplot/robo_plotter.pl
```

2) Main Perl program – robo_plotter.pl

```
#!/opt/apps/perl_5.6.1/bin/perl
#
# robo_plotter.pl  Chethan Parameswariah First release Sep 12 2003 Version 1.0
#
#
#
# ##############################################################################
#
#
#
#
# Need these lib modules - this prepends to @INC at run time
#
use lib "/opt/apps/perl_5.6.1/modules/HTML-Parser-2.22/blib/lib";
use lib "/opt/apps/perl_5.6.1/modules/libwww-perl-5.42/lib";
use lib "/opt/apps/perl_5.6.1/modules/URI-1.02";
use lib "/opt/apps/perl_5.6.1/modules/HTML-Parser-2.22/lib";
use lib "/opt/apps/perl_5.6.1/modules/MIME-Base64-2.11/blib/lib";

# Tell what modules to use
#
use HTTP::Request::Common qw(POST);
use LWP::UserAgent;
use CGI;

# Set variables
#
$DEBUG = 0;
$AUTO_ELOG=1;
$ifo = "l1";
$science_run = "S3";

#  Get the gps time for today and yesterday
$gps_utc_diff = 13;
$now = time;
$gpstime = $now - 315964800 + $gps_utc_diff;
$yesterday_gpstime = $gpstime - 86400;
$lastweek_gpstime = $gpstime - (86400 *7);

if ($DEBUG) {
    print "gpstime = $gpstime\n";
    print "Yesterday's gpstime = $yesterday_gpstime\n";
    print "Last week's gpstime = $lastweek_gpstime\n";
}


# This defines who elogs it
$username = "cparames";
$password = "wave\$";

# Get hour, today's date, month and year
($HOUR_NUMBER, $DAY_NUMBER, $MONTH_NUMBER, $YEAR_NUMBER) = (localtime(time)) [2,3,4,5];
$MONTH_STRING = (qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec))[(localtime) [4]];
$MONTH_NUMBER += 1;
   $MONTH_NUMBER_x = $MONTH_NUMBER;
if ($MONTH_NUMBER < 10) {
```

```
   $MONTH_NUMBER_x = " ".$MONTH_NUMBER;
}
if ($MONTH_NUMBER < 10) {
   $MONTH_NUMBER = "0".$MONTH_NUMBER;
}
   $DAY_NUMBER_x = $DAY_NUMBER;
if ($DAY_NUMBER < 10) {
   $DAY_NUMBER_x = " ".$DAY_NUMBER;
}
if ($DAY_NUMBER < 10) {
   $DAY_NUMBER = "0".$DAY_NUMBER;
}
$YEAR_NUMBER += 1900;

# Get yesterday's date, month and year
($DAY1_NUMBER, $MONTH1_NUMBER, $YEAR1_NUMBER) = (localtime(time-86400)) [3,4,5];
$MONTH1_STRING = (qw(Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov\
Dec))[(localtime(time-86400)) [4]];
$MONTH1_NUMBER += 1;
   $MONTH1_NUMBER_x = $MONTH1_NUMBER;
if ($MONTH1_NUMBER < 10) {
   $MONTH1_NUMBER_x = " ".$MONTH1_NUMBER;
}
if ($MONTH1_NUMBER < 10) {
   $MONTH1_NUMBER = "0".$MONTH1_NUMBER;
}
   $DAY1_NUMBER_x = $DAY1_NUMBER;
if ($DAY1_NUMBER < 10) {
   $DAY1_NUMBER_x = " ".$DAY1_NUMBER;
}
if ($DAY1_NUMBER < 10) {
   $DAY1_NUMBER = "0".$DAY1_NUMBER;
}
$YEAR1_NUMBER += 1900;

if ($DEBUG) {
print "Hour = $HOUR_NUMBER\tMonth = $MONTH_STRING - $MONTH_NUMBER\tDay =\
$DAY_NUMBER,\tYear = $YEAR_NUMBER\n";
print "Yest Hour = $HOUR1_NUMBER\tYest Month = $MONTH1_STRING -
$MONTH1_NUMBER\tYest Day = $DAY1_NUMBER,\tYest Year = $YEAR1_NUMBER\n";
}

#use this if you want to test the script for a day when reboot was made, change
the value
#$DAY_NUMBER = 3;

# Generate the daily summary plot and save it as a ps file
$dvcommandpath = "/cvs/cds/project/roboplot";
$daily_stat_dir = "/opt/LLO/c/ops/public_html/$science_run/DailyStatistics";
$daily_sumplot_file =\
"$daily_stat_dir/Summary_".$MONTH1_NUMBER.$DAY1_NUMBER."_$ifo";

$command = "$dvcommandpath/dv_command_file 10.100.0.30 0\
$dvcommandpath/S3DailySummary.xml $yesterday_gpstime 86400 0\
$daily_sumplot_file.ps";

if ($DEBUG) {
       print "$command \n";
}
system($command);

# Wait for 10 seconds for the file to be saved and convert to pdf.
sleep 10;
$command = "/opt/apps/gnu/bin/ps2pdfL $daily_sumplot_file.ps\
$daily_sumplot_file.pdf";
if ($DEBUG) {
       print "$command \n";
}
system($command);
sleep 10;

# Make the elog entry finally
# comment this out if you want to elog
#$AUTO_ELOG = 0;
###### Auto Elog ######
if ($AUTO_ELOG) {

# Create a new user agent

$ua = LWP::UserAgent->new();
```

```
# Since I need a proxy at LLO, set the proxy here
#
$ua->proxy('http','http://london.ligo-la.caltech.edu:80/');


# set url of elog
#
my $URL = 'http://www.ligo-la.caltech.edu/ilog/pub/ilog.cgi?';

# Elog file date
#
$log_file_date = "$MONTH_NUMBER/$DAY_NUMBER/$YEAR_NUMBER";
#$log_file_date = "12/23/2001";

$comment_string_header = "<H2> <font color=\"green\">Daily Summary Plot from\
$MONTH1_STRING/$DAY1_NUMBER/$YEAR_NUMBER - $MONTH_STRING/$DAY_NUMBER/$YEAR_NUMBER\
</font></H2>";
$comment_string_footer = "<P>This entry automatically elogged by <b>ROBO PLOTTER\
</b><BR>";
$comments = $comment_string_header.$comment_string.$comment_string_footer;

if ($DEBUG) {
        print $comments."\n";
}

# POST it with contents and values
#
my $request = POST $URL,
        Content_Type => 'multipart/form-data',
        Content =>
        [
         group => 'detector',
         task => 'makeEntry',
         log_file_date => $log_file_date,
         comments => $comments,
         keywords => 'CDS',
         priority => 'normal',
        'image_to_include' => ["$daily_sumplot_file.pdf"],
         entry_author => 'cparames',
         'submit' => 'Submit Log Entry'
        ];

# Dont forget the authorization
#
$request->authorization_basic($username, $password);

# And finally make the call.
$content = $ua->request($request)->as_string;

if ($DEBUG) {
print $content;
};
}

# Make a entry that tells the program ran successfully.
$statuslogfile = "/cvs/cds/llo/logs/roboplotterlog.html";

  # finally log we have run
  open(LOG,">$statuslogfile")||die "Cannot open $statuslogfile\n";
  print LOG "<HTML><BODY>\n";
  print LOG "<H4>Robo-plotter ran successfully </H4> \n";
  print LOG " <HR>\n";
  print LOG "Completed at  ".returnTimeStamp()."\n";
  print LOG "</BODY></HTML>\n";
  close LOG;


# ------------------------------------------------------------------------
#
# Subroutine returnTimeStamp()
#
# Returns the current time stamp as a string
#
# ------------------------------------------------------------------------
sub returnTimeStamp {

  @timestamp = localtime(time);
  $thisday = (Sun,Mon,Tue,Wed,Thu,Fri,Sat)[$timestamp[6]];
  $MONTH_NUMBER = $timestamp[6] + 1;
  $thismonth = (Jan,Feb,Mar,Apr,May,Jun,Jul,Aug,Sep,Oct,Nov,Dec)[$timestamp[4]];
```

```
  $MONTH_NUMBER = $timestamp[4] + 1;
  if ($MONTH_NUMBER < 10){
    $MONTH_NUMBER = "0".$MONTH_NUMBER;
  }
  $DAY_NUMBER = $timestamp[3];
  if ($DAY_NUMBER < 10){
    $DAY_NUMBER = "0".$DAY_NUMBER;
  }
  # Y2k stuff
  # Year 2000 defined as $timestamp[5] = 100
  if($timestamp[5]> 99) {
    $timestamp[5] = $timestamp[5] - 100;
    $thisyear = "200".$timestamp[5];
  } else {
    $thisyear = "19".$timestamp[5];
  }
  $YEAR_NUMBER = $thisyear;
  if ($timestamp[2] < 10) {
    $thishour = "0".$timestamp[2];
  } else {
    $thishour = $timestamp[2];
  }
  if ($timestamp[1] < 10) {
    $thismin = "0".$timestamp[1];
  } else {
    $thismin = $timestamp[1];
  }
  if ($timestamp[0] < 10) {
    $thissec = "0".$timestamp[0];
  } else {
    $thissec = $timestamp[0];
  }

  $thisdate = $timestamp[3];

  return "$thisday$thisdate$thismonth$thisyear-$thishour:$thismin:$thissec";


}
```

3) Configuration file – S3DailySummary.xml

```
9
0
L1:LSC-AS_DC
no_conv
0
16384
L1:LSC-LA_SPOB_NORM
no_conv
0
300
L1:LSC-LA_PTRX_NORM
no_conv
1000
2000
L1:LSC-Range_kpc_CalLine
no_conv
0
3000
L1:DMT-LOCKLOSS_StateVEC
no_conv
0
5
L1:LSC-AS_Q_A1-927.70
no_conv
0
0.03
L1:LSC-AS_Q_DQGlitch_4_100
no_conv
0
30
L0:PEM-EX_SEISY_0.1-0.3Hz
no_conv
0
100000
L0:PEM-EY_SEISY_1-3Hz
no_conv
0
```

```
2000
1
60
1
1
1
```

4) README file – Details on the configuration file

```
Commandline version of Grace-Dataviewer for playback
====================================================

$DVPATH must be set as same as Dataviewer (Grace).

dv_command takes six arguments:
  Server IP, Server Port, Input File, Starting time (GPS), Duration (in seconds),
Conversion (1-yes 0-no)

E.g.,  dv_command 198.129.208.138 0 playset 741824155 300 1

dv_command_file, which saves the Grace output as a postscript file
instead of showing it on screen, takes an additional argument:
  Server IP, Server Port, Input File, Starting time (GPS), Duration (in seconds),
Conversion (1-yes 0-no), Output (ps) file

E.g.,  dv_command_file 198.129.208.138 0 playset 741824155 300 1 dv.ps


Input File Format (do not include the comments):

Total channels N            // integer <= 16
auto                        // 1 (auto setting) or 0 (non-auto)
Ch.1 name
Ch.1 unit
Ch.1 y-min                  // float; only when auto = 0
Ch.1 y-max                  // float; only when auto = 0
...
Ch.N name
Ch.N unit
Ch.N y-min                  // float; only when auto = 0
Ch.N y-max                  // float; only when auto = 0
X-axis format               // 1 (GPS) or 0 (UTC)
decimation                  // 0 (full data), 1 (sec trend), 60 (min trend) or 10
(10 min trend)
mean                        // 1 (display) or 0 (don't display)
max                         // 1 (display) or 0 (don't display)
min                         // 1 (display) or 0 (don't display)


Example of Input File:

8
1
C1:PSL-FSS_MIXERM_F
volts
C1:PSL-FSS_PCDRIVE_F
volts
C1:PSL-FSS_SLOWDC_F
volts
C1:PSL-FSS_FAST_F
volts
C1:PSL-ISS_ISERR_F
volts
C1:PSL-ISS_ISS_ACTM_F
volts
C1:PSL-PMC_ERR_F
volts
C1:PSL-PMC_PZT_F
volts
1
1
1
1
1
```