Photometry of Eclipsing Binary V11 in NGC 7789

Joey Mckenzie
Department of Astronomy
San Diego State University

August 9, 2016

Contents

1 Introduction

1.1 Background and Motivation

Age determination for stars plays a key role in the science of astronomy as it lends astronomers an insight into how the very stars we observe in the night sky live their lives. Stars are born in massive molecular clouds, coined stellar nurseries, in various regions from within the cloud where material is dense enough to achieve star forming conditions. By understanding how the timescale on which a star lives its life (alongside other physical characteristics such as mass and size), its evolutionary track may be inferred and how its post main-sequence journey will ultimately determine the star's fate. If a star's mass is known, an estimate for the lifespan of a star can be made due to an observed empirical relation between mass and lifetime; as astronomers understand, bigger, more massive stars observed on the main-sequence will live much shorter lives than their low mass counterparts. While a single star may prove itself to be rather difficult to have its age determined by dating techniques currently available, star clusters are of particular interest to astronomers as they provide ideal conditions for stellar chronology research due to similar composition and age relative to other stars in the cluster.

Age for a star, as previously mentioned, allows astronomers to predict how a star will end its life. Low and intermediate mass stars will spend the majority of their life on the main-sequence, as all stars do, until they have exhausted all their hydrogen fuel in their cores and begin fusing heavier elements. Once this late stage is reached, evolution off the main sequence takes place, where these stars will increase their size greatly to become red giants, spending most of the rest of their lives pursuing various transformations within this branch. Once these low and intermediate mass stars exhaust their stay in the giant branch, they will eventually settle into their final stage of evolution, becoming white dwarfs while maintaining stability through electron degeneracy pressure supporting the white dwarf and remaining in this stage for longer than the age of the known universe. High mass stars, on the other hand, will live the majority of their lives on the main sequence, like their low/intermediate mass counterparts, but will evolve earlier off the main sequence into their giant phases. These high mass stars will evolve into supergiant stars, and depending on how massive the star is, will eventually supernova and collapse into neutron stars, or for the most massive of stars, black holes.

1.2 Age Dating Techniques

Of the handful of useful age dating techniques currently in use for stars, the primary method used for my project, using eclipsing binaries in open cluster NGC 7789, and a select few others (gyrochronology, isochrone fitting on the H-R diagram, and white dwarf cooling age [1])[1] will be discussed. Through the use of observed eclipsing binaries in star clusters, the age of the cluster itself may be inferred. By observing binary stars within crowded fields, light curves may be generated through the use of photometric methods. Treating each star as point source of light allows for the use of fitting a point-spread function (PSF) for each star. By invoking the use of PSF photometry for crowded fields, fitting functions reveal certain binaries within the cluster (as in the case of V11 in NGC 7789) by their magnitude "dip" as an eclipse is observed. Measuring these dips and creating light curves that pronounce the features of the curve allow further information about the binary to be extracted. For example, these light curves carry information about the orbital period of the binary pair, from which the mass can be inferred. Measuring the spectra of binary stars produces spectral lines that may appear to be red or blue shifted, and thus the radial velocity of each component may measured as well. Using the effective temperature, T_{eff}, of the binary by measuring the energy flux of each star at their surfaces as well as the observed color of the binaries, each can be fit on a

[1]Though Soderblom discusses several different age dating techniques, I focused on these three, alongside the use of eclipsing binaries for my research.

color-magnitude diagram (CMD), and under the assumption that all the stars in the cluster are of approximately the same age, the age can be determined by estimation of the turnoff point, or hinge pattern formed on the diagram indicating the main sequence stars beginning their evolutionary journey to the later stages of life.

Another method that works rather well with the previously mentioned technique employing the use of eclipsing binaries is isochrone fitting for main sequence stars. The Hertzsprung-Russell diagram (HRD) is one of stellar astronomy's most powerful tools, relating the luminosity and temperature of stars; however, these quantities are rather difficult to measure to desired precision, and are rather substituted for color (temperature) and magnitude (luminosity). The CMD for star clusters, as mentioned previously, serves for a starting point for this method, where isochrone models are fit. Using a star's temperature and luminosity, alongside its metallicity[2], placement of a model star taken from a particular study may be fit to a CMD. Its placement on the CMD amongst the hundreds to thousands of other star allows for the presumptive use of sophisticated statistical models of how the star will (possibly) evolve. [1] By following the stars from the cluster on their respective isochrones through their evolution on the CMD through the use of computer-generated models, the age of the cluster may be constrained due to the fit of many stars from the object itself and the assumption of age equivalence.

Gyrochronology presents another useful technique for age determination of binary stars, in that being the observed relationship between mass and rotation period. The rotation of a star has long been understood to deterministically slowdown as star ages, providing astronomers with yet another possible tool to estimate how old a star is from the characteristics of its spindown. To measure a star's rotation requires intensive observation, however, at the cost of obtaining precise ages for solar type stars. An empirical relation can be calibrated between

[2]Although I hand wave by remarking that these measurements are used, credit must be given to those stellar astronomers that go through extraordinary efforts to obtain these quantities.

the rotation period and color of a star to match it to an estimated age. This method was conducted using low mass stars, however, and is still rather ambiguous for high mass stars ($M > 1.4M_{\odot}$).

White dwarfs, or what remains of a star after it has completed its main sequence and post-main sequence lifetimes, provide yet another method of cluster age dating. The temperature of a white dwarf remains roughly constant over large timescales. However, shortly after the birth of a white dwarf, the stellar remnant cools off rapidly relative to its cooling at later stages of its life; herein lies the primary idea for the white dwarf cooling age estimate [2]. By measuring the luminosity of a white dwarf, a rather difficult feat in itself due to being some of the faintest known stars, alongside its mass, a cooling timescale may be inferred as the luminosity of the white dwarf slowly fades due to the apparent cooling of the star. By comparing models of cooling ages for white dwarfs and theoretical models (pure carbon-oxygen WDs), a relationship may be calibrated.

1.3 Why Age Matters

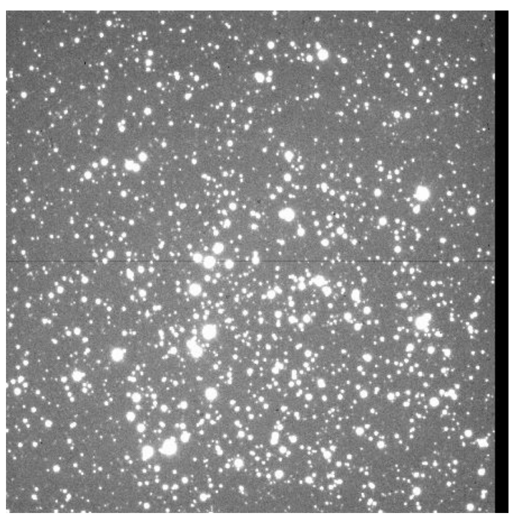
We see the importance of stellar ages all throughout astronomy, as to understand the age of a star is to understand fundamentally how that star will live its life. Stellar evolution models will be further refined as more work is done in the field of stellar age dating. Continuing to improve upon existing techniques, such as the use of eclipsing binaries, and exploration into newer methods, e.g. asteroseismology, will prove pivotal for the science as a whole as we begin to apply stellar ages to other branches of astronomy, an example being how a star's age can help to understand how orbiting planets in its system interact with one another. What I hope to accomplish with this project is the successful photometry of eclipsing binaries in NGC7789 and generation of light curves to obtain useful parameters in an attempt to estimate an age for the cluster, in hopes of furthering the science of stellar chronology.

2 Photometry of NGC 7789

In the initial reduction of data for NGC 7789 [3] located in the constellation of Cassiopeia, images were taken over a span of 11 separate nights.[3] Each night's photometry was performed in a similar manner through the use of IRAF and Peter B. Stetson's program DAOPHOT, specifically designed for crowded-field stellar photometry. The resultant frames were then analyzed by several codes written by myself, as well as the assistance of the light curve information extraction code provided by Eric Sandquist. Four filters in total were observed: I, B, V, and R. Each night was separated into master filter directories and photometry was performed using all frames of a particular filter for the final generation of the light curves below.

2.1 Bias Trim

To begin, each image had its bias strip trimmed, resultant from CCD 2005 used by the Mt. Laguna Observatory telescope for data acquisition.



[3]Obtained by Eric Sandquist, Professor of Astronomy at San Diego State University.

We see the raw image containing all the peculiarities of CCD 2005, namely the bad pixel row roughly in the center of frame as well as the bias strip to the right of the field. Using the IRAF task `ccdproc`, each frame was trimmed to leave behind just the field using the following parameters:

```
                                                   I R A F
                                    Image Reduction and Analysis Facility
PACKAGE = ccdred
   TASK = ccdproc

images  =           @o112.list  List of CCD images to correct
(output =         @o112s.list) List of output CCD images
(ccdtype=                    ) CCD image type to correct
(max_cac=                   0) Maximum image caching memory (in Mbytes)
(noproc =                  no) List processing steps only?

(fixpix =                  no) Fix bad CCD lines and columns?
(oversca=                 yes) Apply overscan strip correction?
(trim   =                 yes) Trim the image?
(zerocor=                  no) Apply zero level correction?
(darkcor=                  no) Apply dark count correction?
(flatcor=                  no) Apply flat field correction?
(illumco=                  no) Apply illumination correction?
(fringec=                  no) Apply fringe correction?
(readcor=                  no) Convert zero level image to readout correction?
(scancor=                  no) Convert flat field image to scan correction?

(readaxi=                line) Read out axis (column|line)
(fixfile=                    ) File describing the bad lines and columns
(biassec=   [2100:2200,2:2047]) Overscan strip image section
(trimsec=     [22:2064,2:2047]) Trim data section
(zero   =                    ) Zero level calibration image
(dark   =                    ) Dark count calibration image
(flat   =                    ) Flat field images
(illum  =                    ) Illumination correction images
(fringe =                    ) Fringe correction images
(minrepl=                  1.) Minimum flat field value
(scantyp=           shortscan) Scan type (shortscan|longscan)
(nscan  =                   1) Number of short scan lines

(interac=                  no) Fit overscan interactively?
(functio=           chebyshev) Fitting function
(order  =                  50) Number of polynomial terms or spline pieces
(sample =                   *) Sample points to fit
(naverag=                   1) Number of sample points to combine
(niterat=                   1) Number of rejection iterations
(low_rej=                  3.) Low sigma rejection factor
(high_re=                  3.) High sigma rejection factor
(grow   =                  0.) Rejection growing radius
(mode   =                  ql)
```

The 100 column strip, once removed from the frame, left behind just the field, which allowed for subtraction of the bias frames from each image. Compiling all images from a night into a single list, exemplified above as `o112.list`, allowed for the use of the `sed` command for creating a new list of similar images containing file tages related to the IRAF procedures performed on the frames. The tags, as will be shown throughout the succeeding steps, are as follows:

- `s` = bias strip subtracted
- `b` = bias subtracted from each frame
- `c` = corrected for nonlinearity of CCD 2005
- `f` = flat field divided out of each frame

Using `sed` in the following manner,

```
ccdred> !sed 's/\.fits/s\.fits/g' o112.list > o112s.list
```

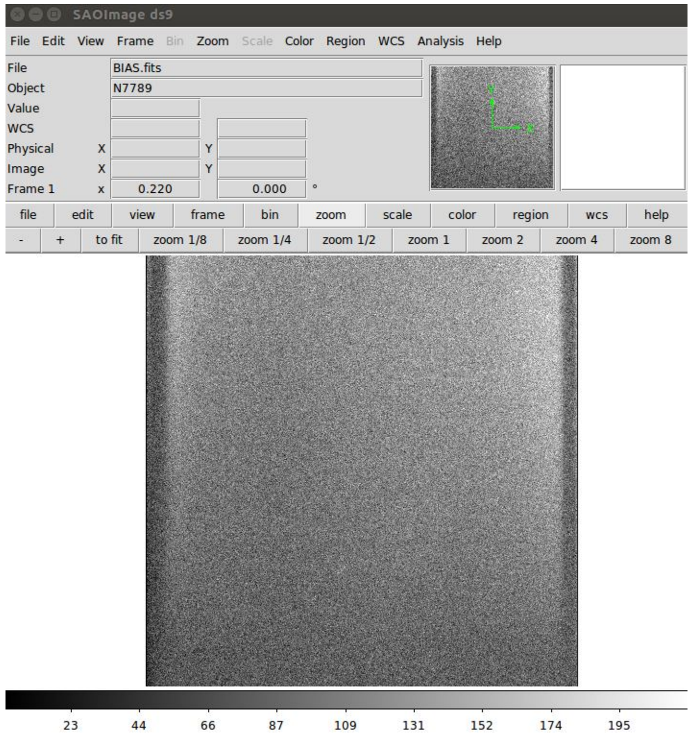
files were easily created with descriptive tags to allow for distinction from the original frames.

2.2 Bias Subtraction

With the bias strip subtracted from each image, as well as sorting newly created frames into a directory of the similar file descriptor, the next step in the photometry of NGC 7789 was the bias/background noise subtraction from each frame. Combining all bias images from a single night into a simple list and running the IRAF command `imcomb` using the parameters,

```
                                              I R A F
                                    Image Reduction and Analysis Facility
PACKAGE = immatch
   TASK = imcombine

input   =             @bias.list  List of images to combine
output  =                   BIAS  List of output images
(headers=                      ) List of header files (optional)
(bpmasks=                      ) List of bad pixel masks (optional)
(rejmask=                      ) List of rejection masks (optional)
(nrejmas=                      ) List of number rejected masks (optional)
(expmask=                      ) List of exposure masks (optional)
(sigmas =                      ) List of sigma images (optional)
(imcmb  =                    $I) Keyword for IMCMB keywords
(logfile=                STDOUT) Log file

(combine=                median) Type of combine operation
(reject =                minmax) Type of rejection
(project=                    no) Project highest dimension of input images?
(outtype=                  real) Output image pixel datatype
(outlimi=                      ) Output limits (x1 x2 y1 y2 ...)
(offsets=                  none) Input image offsets
(masktyp=                  none) Mask type
(maskval=                     0) Mask value
(blank  =                    0.) Value if there are no pixels

(scale  =                  none) Image scaling
(zero   =                  mode) Image zero point offset
(weight =                  none) Image weights
(statsec=                      ) Image section for computing statistics
(expname=                      ) Image header exposure time keyword

(lthresh=                 INDEF) Lower threshold
(hthresh=                 INDEF) Upper threshold
(nlow   =                     0) minmax: Number of low pixels to reject
(nhigh  =                     1) minmax: Number of high pixels to reject
(nkeep  =                     1) Minimum to keep (pos) or maximum to reject (neg)
(mclip  =                   yes) Use median in sigma clipping algorithms?
(lsigma =                    3.) Lower sigma clipping factor
(hsigma =                    3.) Upper sigma clipping factor
(rdnoise=                    0.) ccdclip: CCD readout noise (electrons)
(gain   =                    1.) ccdclip: CCD gain (electrons/DN)
(snoise =                    0.) ccdclip: Sensitivity noise (fraction)
(sigscal=                   0.1) Tolerance for sigma clipping scaling corrections
(pclip  =                  -0.5) pclip: Percentile clipping parameter
(grow   =                    0.) Radius (pixels) for neighbor rejection
(mode   =                    ql)
```

to create a master bias image yielded the following:



The master bias frame was then subtracted out of the remaining frames using `imarith`, creating new frames with the `b` descriptor to differentiate from the previous frames using `sed`.

2.3 Correction for Nonlinearity in CCD 2005

Due to a discovery in the response of CCD 2005 nonlinear in nature [4], each night had its flat field and science frames corrected as a result. After the bias subtraction, image lists were made once more, and using `sed`, duplicated with the insertion of the `c` descriptor onto each flat field and science frame. The nonlinear corrections were made using the IRAF task `irlincor` located in the in the `irred` package. The parameters used for each coefficient of the correction for `irlincor` are as follows:

```
                                                               I R A F
                                             Image Reduction and Analysis Facility
PACKAGE = irred
   TASK = irlincor

input   =         @s212sb.list  Input  images
output  =        @s212sbc.list  Output images
(section=                    ) Image section to correct
(coeff1 =              1.01353) First  coefficient of correction equation
(coeff2 =            -0.115576) Second coefficient of correction equation
(coeff3 =            0.0296378) Third  coefficient of correction equation
(mode   =                   ql)
```

The nonlinear response was recognized in November of 2012 by several SDSU astronomers while performing photometry on data taken from multiple Landolt fields. The photometry revealed unusual trends in data between measured band magnitudes and the difference between the measured magnitude and the best-fit model of the magnitudes (residuals). In turn, this prompted SDSU astronomers Douglas Leonard and Bob Leach to investigate these unusual responses produced by CCD 2005, leading to the conclusion of nonlinearity in response of the aforementioned light gathering device. Corrected frames for NGC 7789 in my data are designated by the `c` file descriptor, once again made using the `sed` command.

2.4 Flat Field

Once images (flat field and science frames) had been corrected for nonlinear response, flat field images pertaining to their specific band filters had their effects on the science frames divided out. By combining flat field images according to band in the following manner,

```
irred> ls *idflat* > idflats.list
irred> ls *vdflat* > vdflats.list
```

and piping all flat field images into separate lists according to band pass such as,

```
irred> ls o112n7789i* > o112i.list
irred> ls o112n7789v* > o112v.list
irred> !sed 's/\.fits/f\.fits/g' o112i.list > o112if.list
irred> !sed 's/\.fits/f\.fits/g' o112v.list > o112vf.list
```

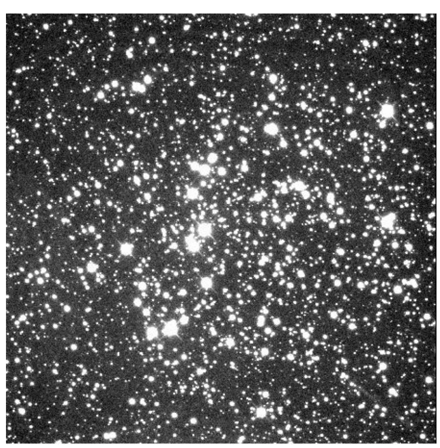
allowed for the use of `imcomb` once more for the flat field image combination using the paramters:

```
                                                      I R A F
                                        Image Reduction and Analysis Facility
PACKAGE = immatch
   TASK = imcombine

input   =        @idflats.list  List of images to combine
output  =               IDFLAT  List of output images
(headers=                     ) List of header files (optional)
(bpmasks=                     ) List of bad pixel masks (optional)
(rejmask=                     ) List of rejection masks (optional)
(nrejmas=                     ) List of number rejected masks (optional)
(expmask=                     ) List of exposure masks (optional)
(sigmas =                     ) List of sigma images (optional)
(imcmb  =                   $I) Keyword for IMCMB keywords
(logfile=               STDOUT) Log file

(combine=               median) Type of combine operation
(reject =               minmax) Type of rejection
(project=                   no) Project highest dimension of input images?
(outtype=                 real) Output image pixel datatype
(outlimi=                     ) Output limits (x1 x2 y1 y2 ...)
(offsets=                 none) Input image offsets
(masktyp=                 none) Mask type
(maskval=                    0) Mask value
(blank  =                   0.) Value if there are no pixels

(scale  =                 mode) Image scaling
(zero   =                 none) Image zero point offset
(weight =                 none) Image weights
(statsec=                     ) Image section for computing statistics
(expname=                     ) Image header exposure time keyword

(lthresh=                INDEF) Lower threshold
(hthresh=                INDEF) Upper threshold
(nlow   =                    0) minmax: Number of low pixels to reject
(nhigh  =                    1) minmax: Number of high pixels to reject
(nkeep  =                    1) Minimum to keep (pos) or maximum to reject (neg)
(mclip  =                  yes) Use median in sigma clipping algorithms?
(lsigma =                   3.) Lower sigma clipping factor
(hsigma =                   3.) Upper sigma clipping factor
(rdnoise=                   0.) ccdclip: CCD readout noise (electrons)
(gain   =                   1.) ccdclip: CCD gain (electrons/DN)
(snoise =                   0.) ccdclip: Sensitivity noise (fraction)
(sigscal=                  0.1) Tolerance for sigma clipping scaling corrections
(pclip  =                 -0.5) pclip: Percentile clipping parameter
(grow   =                   0.) Radius (pixels) for neighbor rejection
(mode   =                   ql)
```

Running `imcomb` on the flat field images produced a master flat field image for each band filter. However, each master flat had its ADU count normalized before being divided out of the remaining science images. The normalization of the flat field master images allowed for counts to remain relatively unchanged after flat field division, as to not distort counts for further photometric data analysis. The flat field image combination was performed using `imarith` and `imhead` to obtain the average number of counts per pixel to perform the normalization of the master flat field image in the following fashion:

```
irred> imcomb
List of images to combine (@idflats.list):
List of output images (IDFLAT):

May 10 23:35: IMCOMBINE
  combine = median, scale = mode, zero = none, weight = none
  reject = minmax, nlow = 0, nhigh = 1
  blank = 0.
                Images     Mode  Scale
   o112idflat01sbc.fits  22212.  1.000
   o112idflat02sbc.fits  22012.  1.009
   o112idflat03sbc.fits  21774.  1.020
   o112idflat04sbc.fits  21783.  1.020
   o112idflat05sbc.fits  21487.  1.034
   o112idflat06sbc.fits  21766.  1.021
   o112idflat07sbc.fits  21796.  1.019
   o112idflat08sbc.fits  21741.  1.022
   o112idflat09sbc.fits  21951.  1.012
   o112idflat10sbc.fits  22024.  1.009
   o112idflat11sbc.fits  21961.  1.011
   o112idflat12sbc.fits  21737.  1.022
   o112idflat13sbc.fits  21929.  1.013
   o112idflat14sbc.fits  21894.  1.015
   o112idflat15sbc.fits  21635.  1.027
   o112idflat16sbc.fits  21503.  1.033
   o112idflat17sbc.fits  21483.  1.034
   o112idflat18sbc.fits  21475.  1.034
   o112idflat19sbc.fits  21498.  1.033
   o112idflat20sbc.fits  21403.  1.038

  Output image = IDFLAT, ncombine = 20
irred> imstat IDFLAT.fits
#               IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
          IDFLAT.fits   4116176    22191.     379.1    21054.    23316.
irred> imarith IDFLAT.fits / 22191. IDFLAT.fits
irred> imstat IDFLAT.fits
#               IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
          IDFLAT.fits   4116176        1.   0.01708    0.9488     1.051
```

Once the flat field master images were normalized, `imarith` was performed once again on the remaining field frames to produce final images, designated with an `f` file descriptor alluding to flat field division being performed. What was left was the crisp, fully reduced data frame of NGC 7789 in its various band passes.



2.5 Point-Spread Function Photometry

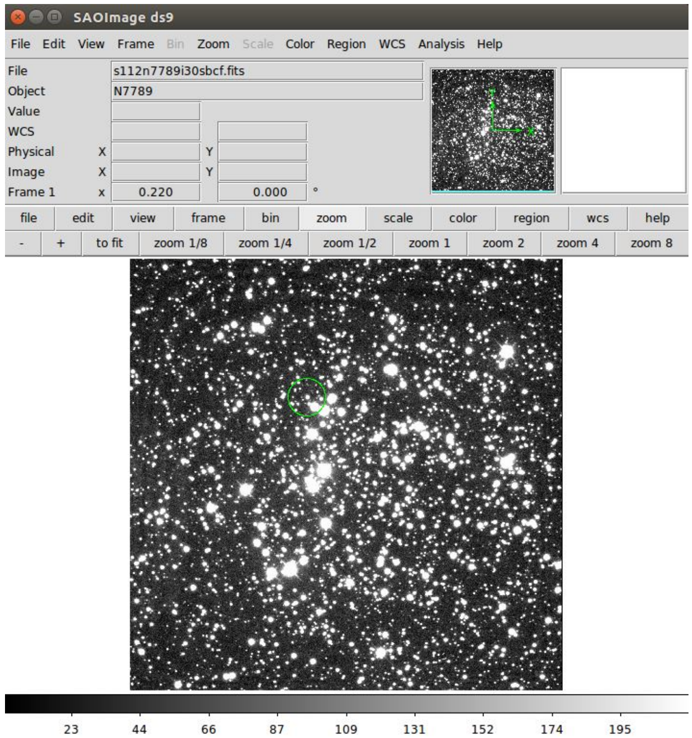
Next in the photometry of NGC 7789 was the fitting of individual point spread functions of stars in the final reduced frames. As stars in the field appear as almost perfect point sources of light, the use of the point spread function to further analyze the data of NGC 7789 and other crowded fields was used. To perform the PSF photometry on the fully reduced nights of data, Peter B. Stetson's standalone software DAOPHOT [5] was used to further data taken from the various nights to produce PSF data for each frame, compiled alongside with ALLSTAR. To perform the PSF photometry on each night using the DAOPHOT standalone script program, DOPSF, the following simple script was written to prepare for DAOPHOT use:

```
        program psf

        implicit none
        integer :: i,v

        open ( unit=10, file='o112i.sh', status='unknown' )
        open ( unit=20, file='o112v.sh', status='unknown' )

        write(10,1)
        write(20,1)
1       format('#!/bin/bash')


        DO i = 1,30

                write(10,2) i
2               format('./dopsf o112n7789i',i2.2,'sbcf 3 5')

        END DO

        DO v = 1,57

                write(20,3) v
3               format('./dopsf o112n7789v',i2.2,'sbcf 3 5')

        END DO

        STOP

        END PROGRAM psf
```

The purpose of the simple setup program was to produce an executable script to run DOPSF on each frame in the formatted fashion, with 5 being the PSF photometry type performed on each frame, and 3 being the pixel fitting radius for each converging PSF for each star in question in the image. Running the script produced multiple files for each frame, with the photometry data stored in the files produced by ALLSTAR in the DOPSF script (.als files).

2.6 MATCH and MASTER

Once .als files were created and moved to further subdirectories for organizational purposes, the programs MATCH and MASTER were used to align each frame's stellar coordinates with a chosen reference frame from a given night to allow for further analysis of the images. In the fitting criteria for choosing a reference frame to align subsequent frames to, the chosen frame was an image taken later in the night for better resolution alongside no visible distortion of eclipsing binary V11 [6]. V11, the main interest of my investigation of NGC 7789, is located roughly in the cluster as shown below:



Using the reference frame of each night to serve as a coordinate transformation basis, a script[4] was created to compile transformation coordinates for all stars with PSF fits in each frame. My simple code `filematch` was used to quickly create executable scripts to match frames relative to the reference frame:

```
         PROGRAM match_file

         IMPLICIT NONE
         INTEGER :: v

         OPEN ( UNIT=10, FILE='o211match.sh', STATUS='UNKNOWN' )

         write(10,99)
99       format('#!/bin/bash',/,'./match <<END',/,'o211n7789i50sbc.als',/,'o211n7789i50sbc.mch')

         DO v = 1,101

           write(10,100) v
100        format('o211n7789i',i2.2,'sbc.als',/,'y',/,'n')

         END DO

         write(10,101)
101      format(///,'END')

         STOP

         END PROGRAM
```

[4]Provided by Eric Sandquist, SDSU.

```
#!/bin/bash
./match <<END
j111n7789i20sbcf.als
j111n7789i20sbcf.mch
j111n7789i01sbcf.als
y
n
j111n7789i02sbcf.als
y
n
j111n7789i03sbcf.als
y
n
j111n7789i04sbcf.als
y
n
j111n7789i05sbcf.als
y
n
j111n7789i06sbcf.als
y
n
j111n7789i07sbcf.als
y
n
j111n7789i08sbcf.als
y
n
j111n7789i09sbcf.als
y
n
j111n7789i12sbcf.als
y
n
j111n7789i13sbcf.als
y
n
j111n7789i14sbcf.als
y
n
```

Once the match scripts were created and made executable through the use of `chmod +x`, the MASTER program was run on the corresponding .mch file containing the photometry information from each frame's ALLSTAR file. MASTER was used in the following manner:

```
joey@Joey:~/iraf/o112/o112s/o112sb/o112sbc/o112sbcf/i/ials$ ./master

                    File with list of input files: 1-14.mch

 The maximum permissible number of stars in the master list: 119836
      119836       68589        7031          11

   Minimum number, minimum fraction, enough frames: 2, 0.6, 5.
                                    Maximum sigma: 0.05

                  Desired degrees of freedom ---

                     2:  Translations only
                     4:  Translations, rotation, and scale
                     6:  Full, six-constant linear transformation

                                      Your choice: 6
```

- Minimum number: 2
- Minimum fraction: 0.6
- Star to appear in enough frames: 5

Beginning with a fitting radius of 5, MASTER was iteratively run during this point in the program down to a fitting radius of 1 until approximate convergence was made. Due to the plethora of images contained in a select few nights of data, .mch files were broken apart into smaller sections with MASTER run on each section. The corresponding .mag and .mch files produced by MASTER were then compiled into a master magnitude list and match file, `master.mag` and `master.mch` respectively.

2.7 Light Curve File Generation

Once frames had been run through MATCH and MASTER, the Julian Date (JD) was converted to the Heliocentric Julian Date (HJD) for each frame using information from the image header (`imhead`) from IRAF. Before setting the for each frame, `observatory` parameters were set in the following fashion in IRAF:

```
                                                    I R A F
                                        Image Reduction and Analysis Facility
PACKAGE = noao
   TASK = observatory

command = █                set  Command (set|list|images)
obsid   =                  MLO  Observatory to set, list, or image default
images  =            @ai.list  List of images
(verbose=                   no) Verbose output?

(observa=              obspars) Observatory identification
(name   = Mt. Laguna Observatory) Observatory name
(longitu=          116.4266666) Observatory longitude (degrees)
(latitud=                32.84) Observatory latitude (degrees)
(altitud=                1859.) Observatory altitude (meters)
(timezon=                  -8.) Observatory time zone
override=              obspars  Observatory identification
(mode   =                   ql)
```

Running `observatory` then set the appropriate geographic and timezone information of the Mt. Laguna Observatory 40" where the data was taken to each frame. As seen above, the following parameters were set:

- Latitude to 32.84°
- Longitude to 116.4266666°
- Altitude to 1859 m
- UTC to minus 8 hours

Once this task was performed, `setjd` was then edited to allow for proper setting of the dates:

```
                                                    I R A F
                                      Image Reduction and Analysis Facility
PACKAGE = onedspec
   TASK = setjd

images  =         @j111i.list  Images
(observa=             obspars) Observatory of observation
(date   =            date-obs) Date of observation keyword
(time   =                  ut) Time of observation keyword
(exposur=            exp_time) Exposure time keyword
(ra     =                  ra) Right ascension (hours) keyword
(dec    =                 dec) Declination (degrees) keyword
(epoch  =               epoch) Epoch (years) keyword

(jd     =                  jd) Output Julian date keyword
(hjd    =                 hjd) Output Helocentric Julian date keyword
(ljd    =                 ljd) Output local Julian date keyword

(utdate =                 yes) Is observation date UT?
(uttime =                 yes) Is observation time UT?
(listonl=                  no) List only without modifying images?
(mode   =                  ql)
```

Once the dates were properly set for each frame, `hselect` was then used to create information files (designated by .inf) containing the HJD, frame name, airmass, and exposure time for each frame in the following fashion:

```
                                                    I R A F
                                      Image Reduction and Analysis Facility
PACKAGE = imutil
   TASK = hselect

images  =         @s112v.list  images from which selection is to be drawn
fields  = $I,HJD,AIRMASS,EXP_TIME  fields to be extracted
expr    =                 yes  boolean expression governing selection
(missing=               INDEF) Value for missing keywords
(mode   =                  ql)
```

Piping the output of `hselect` into a corresponding .inf file,

```
s112n7789i06sbcf.fits   2456187.64191974       1.636   300.
s112n7789i07sbcf.fits   2456187.64572770       1.609   300.
s112n7789i08sbcf.fits   2456187.64954723       1.583   300.
s112n7789i09sbcf.fits   2456187.66741803       1.474   300.
s112n7789i10sbcf.fits   2456187.67122599       1.454   300.
s112n7789i11sbcf.fits   2456187.67503395       1.434   300.
s112n7789i12sbcf.fits   2456187.68031185       1.408   300.
s112n7789i13sbcf.fits   2456187.68413139       1.390   300.
```

Using the program ENSEMBLE[5] called for a modified version of the .inf file to be fed into the software, alongside a master star list containing star IDs in the .mag files and master .mch

[5]Provided by Eric Sandquist, SDSU.

file containing the star coordinate transformations contained in the .als files for each frame. My simple program `header` allowed for proper formatting of the .inf files generated from the `hselect` command, as well as setting a null value for the seeing:

```
      program header

        character(len=16),dimension(200) :: image
        double precision,dimension(200) :: hjd, airmass, exp_time, seeing
        character(len=20) :: input, output
        integer :: i


        write(*,'("Input file (.inf): "$)')
        read(*,*) input
        write(*,'("Output file (corrected .inf): "$)')
        read(*,*) output

        open (unit=10, file=input, status='old')
        open (unit=20, file=output, status='unknown')


101     i = i + 1
        seeing(i) = float(i)*0.

        read(10,1,end=100) image(i), hjd(i), airmass(i), exp_time(i)
1       format(a16,6x,f16.8,1x,f5.3,1x,f4.0)

        write(20,2) image(i), hjd(i), airmass(i), seeing(i), exp_time(i)
2       format(1x,a16,4x,f14.6,f6.3,f9.5,f5.0)

        i = i + 1

        go to 101

100     continue

        stop

        end program header
```

3 Light Curve Generation

3.1 Preparing Light Curve Files

Running the program allowed for the .inf files to contain the proper format called for by ENSEMBLE, along with setting the seeing for each frame to 0 for this investigation. The output is as follows:

```
 s112n7789i05sbcf    2456187.637290 1.671  0.00000 300.
 s112n7789i06sbcf    2456187.641920 1.636  0.00000 300.
 s112n7789i07sbcf    2456187.645728 1.609  0.00000 300.
 s112n7789i08sbcf    2456187.649547 1.583  0.00000 300.
 s112n7789i09sbcf    2456187.667418 1.474  0.00000 300.
 s112n7789i10sbcf    2456187.671226 1.454  0.00000 300.
```

Once the appropriate values had been set within the .inf file and running ENSEMBLE to generate a master light curve file (designated .curs), I wrote a simple sorting algorithm `variable` to identify the target of interest for my investigation, V11:

```
        program variable_coords

        implicit none
        character(len=20) :: input
        real :: xlow, xhigh, ylow, yhigh
        real, dimension(300000) :: x, y
        integer, dimension(300000) :: starID
        integer :: i

        open ( unit=10, file='possible.dat', status='unknown' )

        write(*,'("Enter master star list: "$)')
        read(*,*) input
        open ( unit=30, file=input, status='old' )
        open ( unit=20, file='junk.dat', status='unknown' )

        write(*,'("Enter x min: "$)')
        read (*,*) xlow
        write(*,'("Enter x max: "$)')
        read (*,*) xhigh

        write(*,'("Enter y min: "$)')
        read (*,*) ylow
        write(*,'("Enter y max: "$)')
        read (*,*) yhigh

        i = 1
1       read(30,2,end=100) starID(i), x(i), y(i)
2       format(1x,i5,1x,f8.3,1x,f8.3)

        IF ( x(i) .GE. xlow .AND. x(i) .LE. xhigh  .AND. &
                y(i) .GE. ylow .AND. y(i) .LE. yhigh ) THEN

                write(*,3) starID(i), x(i), y(i)
3               format(1x,i5,1x,f8.3,1x,f8.3)

        ELSE

                write(20,*) starID(i), x(i), y(i)

        END IF

        i = i + 1

        go to 1

100     continue

        stop

        end program variable_coords
```

The purpose of `variable` is to search for the ID of binary V11 from the master list of stars stored in the .mag file generated by running MASTER. Running the program prompts the user for four inputs:

- A lower bound for the x-coordinate of V11 on the reference frame
- An upper bound for the x-coordinate
- A lower bound for the y-coordinate
- An upper bound for the y-coordinate

The program then runs through the master star list (stored within the compiled .mag files), sorting through stars contained within the x and y boundaries specified by the user, and reports back possible star IDs that matched the region. The program outputs star IDs, with the possibility of reporting back multiple stars due to the program catching stars from other frames aligned within the area of V11. These stars were neglected, and V11 was properly identified by examining the light curve displaying clear signs of eclipsing events occurring. `Variable`'s output is exemplified as follows:

```
ecl> !./variable
Enter x min: 824.
Enter x max: 852.
Enter y min: 1380.
Enter y max: 1402.
  7803  839.662 1390.591
  7803  839.665 1390.618
 50496  830.223 1398.129
  7803  839.688 1390.633
```

Once possible star IDs were obtained for V11, the command `grep -e` was used on the .curs file generated by ENSEMBLE to pipe all possible entries that had matching IDs into a light curve file. For example,

```
grep -e 7803 ensemble11vpsf.curs > 7803.lc
```

where the extension .lc was used to designate the file as a raw light curve file, containing just information of V11 from ENSEMBLE. The file was the then edited to remove entries of star IDs that did not match the possible ID for V11, as `grep` would catch decimal numbers containing a similar sequence of numbers to the ID in one of the parameters for the star. Once the file was edited to contain just information of V11, my code `lightcurve` was run on the .lc file to extract the magnitude and timestamp of V11 to create a graphable light curve for each filter.

```
PROGRAM lightcurve

    IMPLICIT NONE
    DOUBLEPRECISION,DIMENSION(300000) :: timestamp, mag, phase, phase_p1, phase_m1, JD_over_period
    INTEGER, DIMENSION(300000) :: starID
    INTEGER :: i, j, k, temp
    CHARACTER(len=20) :: input, outputJD, output_phase

    DOUBLEPRECISION, PARAMETER :: V11_period = 5.00333
    DOUBLEPRECISION, PARAMETER :: V11_t0 = 2448540.900000
    DOUBLEPRECISION, PARAMETER :: timeshift = 2450000.000000

    write(*,'("Enter light curve file (.lc): "$)')
    read(*,*) input
    write(*,'("Enter light curve data file name (JD): "$)')
    read(*,*) outputJD
    write(*,'("Enter light curve data file name (phase): "$)')
    read(*,*) output_phase

    OPEN ( UNIT=10, FILE='V11.lc', STATUS='OLD' )
    OPEN ( UNIT=20, FILE='V11jd.dat', STATUS='UNKNOWN' )
    OPEN ( UNIT=30, FILE='V11phase.dat', STATUS='UNKNOWN' )
    OPEN ( UNIT=40, FILE='V11phase_p1.dat', STATUS='UNKNOWN' )
    OPEN ( UNIT=50, FILE='V11phase_m1.dat', STATUS='UNKNOWN' )

    i = 1
100 read(10,1,END=200) timestamp(i), mag(i)
1   format(7x,f11.6,2x,f7.4)

        timestamp(i) = timestamp(i) + timeshift

    JD_over_period(i) = (timestamp(i) - V11_t0)/V11_period

    phase(i) = JD_over_period(i) - INT( JD_over_period(i) )
    phase_m1(i) = phase(i) - 1.

    write(20,2) timestamp(i), mag(i)
2   format(1x,f14.6,2x,f7.4)

    write(30,3) phase(i) , mag(i)
    write(50,3) phase_m1(i), mag(i)
3   format(1x,f8.6,2x,f7.4)

    i = i + 1
    GO TO 100

200 continue

    CLOSE(10); CLOSE(20); CLOSE(30); CLOSE(40)

    STOP

END PROGRAM lightcurve
```

`Lightcurve` takes the timestamp read in from the .lc file and converts the HJD to the phase for the binary. The program is a simple sorting code to arrange the light curve data contained with the .curs files created by ENSEMBLE and create graphical outputs through the use of `xmgrace`. Calculating the binary phase,

$$\phi = \frac{\mathrm{HJD} - t_0}{P} - \mathrm{INT}\left(\frac{\mathrm{HJD} - t_0}{P}\right)$$

where

$$t_0 = 2448540.90 \text{ and } P = 5.0033 \text{ days}^6$$

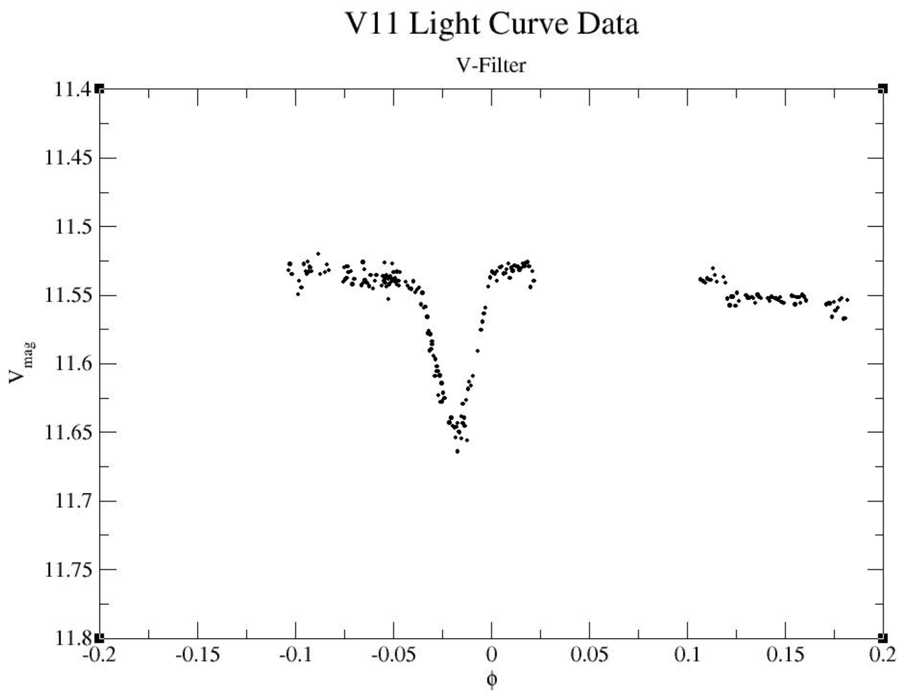
Leaving behind the remaining fractional component of the binary phase, the light curve data for V11 was placed in two separate files, one containing the magnitude and timestamp with the other containing magnitude and phase.

[6]Measurements made by Eric Sandquist, SDSU.

3.2 Light Curves

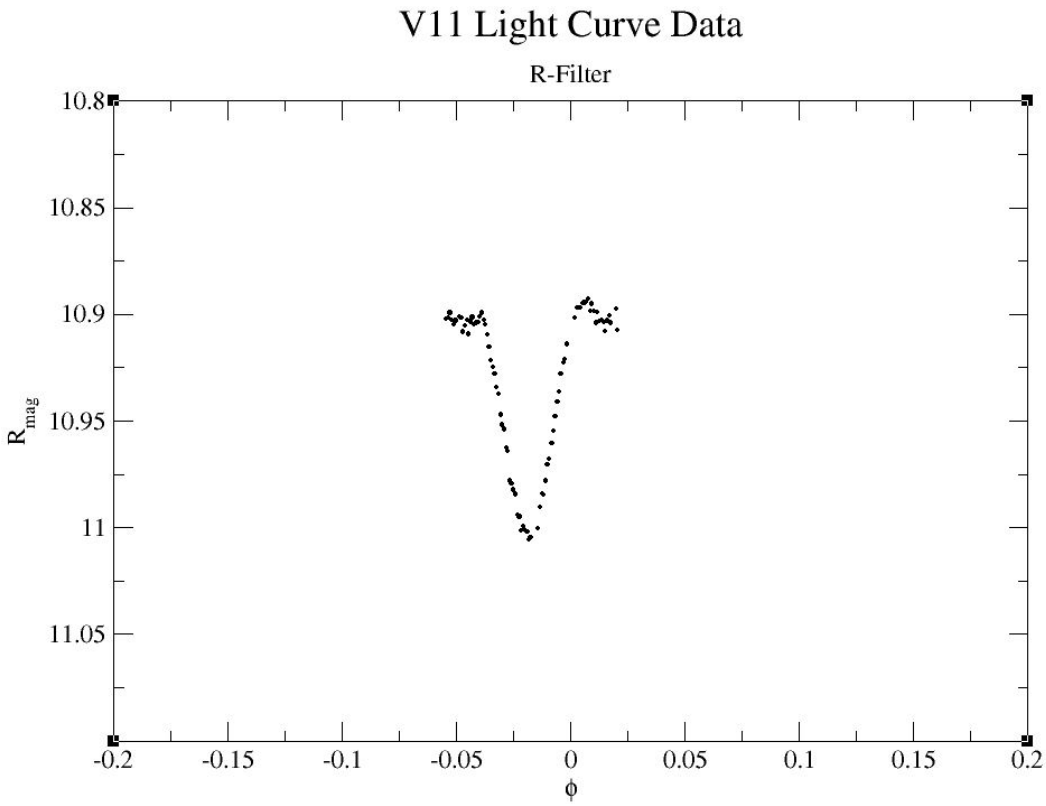
In running through this process for filters I, V, and R, the following light curves were generated:

3.3 V Filter



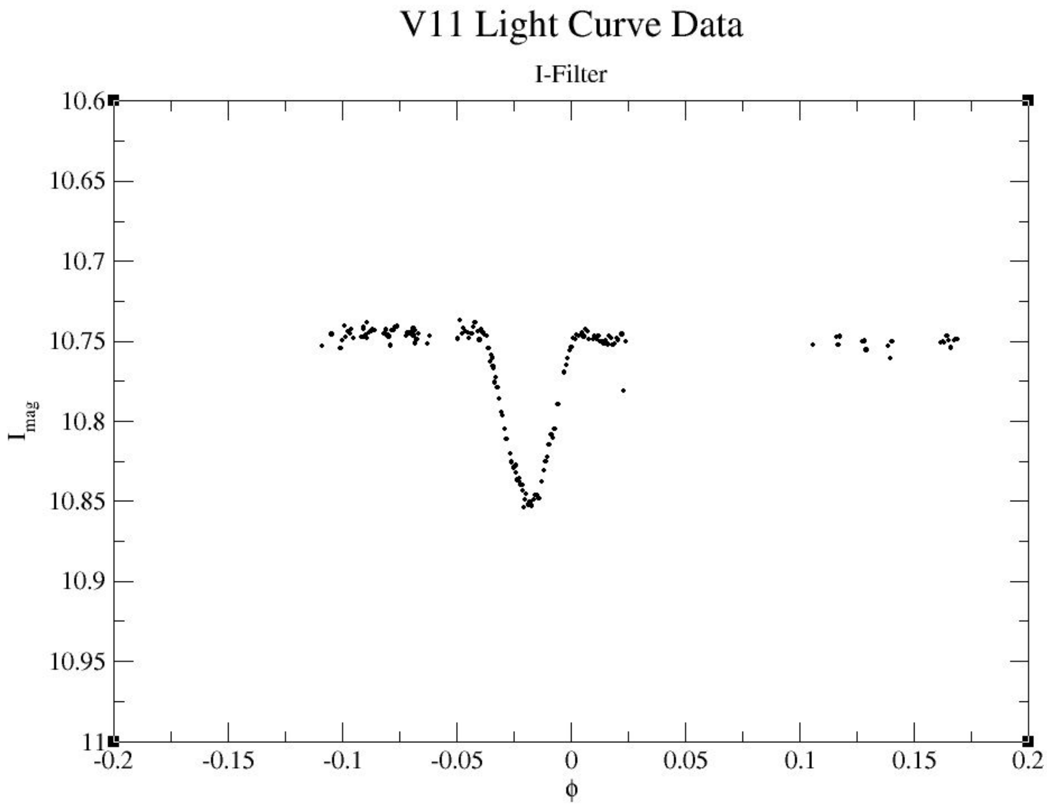
Here we see the characteristic light curve dip, signature of the binary eclipsing. The magnitude of the eclipse appears to drop approximately 0.10 across a span of roughly 0.05 of a phase. Less pronounced than the I band filter, similar characteristics, namely the 0.10 magnitude drop across an approximate 0.05 binary phase occur.

3.4 R Filter



V11 Light Curve Data
R-Filter
10.8
10.85
10.9
10.95
11
11.05
R_{mag}
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15
0.2
ϕ

3.5 I Filter



All data used to generate the light curves above were archived, namely the fully reduced science frames, .als files, and light curve files generated by ENSEMBLE ready for file transfer.

4 Conclusion

Further inspection of data taken from NGC 7789 may be beneficial to obtain more precise information of the V11 binary system beyond the scope of this investigation. With generation of the light curves through the *I*, *R*, and *V* filters, quantitative measurements of the system's parameters, such as the period and radial velocity, may further be refined upon more in depth analysis.

References

[1] D. R. Soderblom, "The Age of Stars," *Annual Review of Astronomy and Astrophysics*, vol. 48, pp. 581–629, may 201.

[2] Hanson *et al.*, "The White Dwarf Cooling Sequence of the Globular Cluster Messier 4," *Astrophysical Journal*, vol. 574, pp. L155–L158, 2002.

[3] B. J. Mochejska and J. Kaluzny, "Variable Stars in the Field of the Open Cluster NGC 7789," *Astrophysical Journal*, vol. 49, pp. 351–370, sep 1999.

[4] D. C. Leonard, "A Non-Linearity in CCD 2005: Discovery, Correction, and Fix," feb 2014.

[5] P. B. Stetson, "DAOPHOT - A Computer Program for Crowded-field Stellar Photometry," *Astronomical Society of the Pacific*, vol. 99, pp. 191–222, Mar. 1987.

[6] X.-B. Zhang, L.-C. Deng, Y. Xin, and X. Zhou, "Searching for Variable Stars in the Field of NGC 7789," *Chinese Journal of Astronomy and Astrophysics*, vol. 3, no. 2, p. 151, 2003.