# The Detection of Black Holes

New Mexico High School
Supercomputing Challenge
Final Report
April 3, 2002

Team #054
Moriarty High School

**Team Members:**
Trevor Brennan
Eric Geib
Jason Gunn
Jens Madsen III
Brian Masters
Eric Owen

**Team Sponsor:**
Paula Avery

**Project Mentor:**
Joshua Barnes
Jens Madsen, Jr.

# Executive Summary

The purpose of this project is to write a C++ program that will simulate the presence of a black hole by observing the changes in the constellations around a simulated black hole. To simulate the detection, we used a method called gravity lensing. We gathered information on stars in the Pleiades Constellation as seen from Earth. We then added the position of the stars to our program and added a black hole at a set location. We used the mass, position and size of the black hole to determine the Einstein Angle of light deflection needed. Our program was made to be user friendly and allows the user to change variables concerning the black hole's position, mass, and distance from Earth. To verify that the program was accurate, we compared our results to those of Joshua Barnes. Our program was an accurate simulation when compared to his data and results. We have concluded that a black hole can greatly alter the position of a star if it is massive enough, but that a small black hole does not have a very significant effect on the stars' positions. Future work includes a GUI interface that can use user-entered variables.

# Table of Contents

## Introduction

The purpose of the project is to write a C++ program that will simulate the presence of a black hole by observing the changes in the constellations around a simulated black hole. We will simulate the detection of the black hole with a technique called gravity lensing. According to Mark O'Brian and John Chang, "Gravity lensing occurs when a black hole passes between a star and the Earth. The black hole acts as a lens when its gravity bends the star's light rays and focuses them on the Earth. From an observer's point of view on the Earth, the star would appear to brighten". (http://www.rdrop.com/users/green/school/detect.htm) We chose this project because it will allow us to simulate the detection of black holes and to let us better understand how a black hole behaves.

### Background

Most, if not all, the bodies in the universe have a gravitational pull. We have escaped some of them while trying to go out in space. We have escaped the earth's gravitational pull and the gravitational pull on the moon and Jupiter. But a black hole is a phenomenon of space. The black hole however has a gravitation pull that is so strong that you can't escape it. In theory, there is no limit to how much mass a black hole can have. According to Chris Miller all black holes contain an 'event horizon'. "The event horizon is a point where you can not return if crossed. When you refer to the size of a black hole, you are talking about the size of the event horizon". Black holes are thought to form from stars or other massive objects if and when they collapse from their own gravity to form an object whose density are infinite or a singularity the more mass the singularity has, the larger the event horizon. No one has ever experienced a black hole, and even if anyone had, they wouldn't live to tell about it because it would cause their body to implode.

(www.eclipse.net/~cmmiler/BH/blkbh.htm)

We found there are three effective ways that have been used to detect black holes. The first one is based on the fact that even after a super-massive star has collapsed and changed into a black hole, the strength of its gravitational pull increases severely, so, anything orbiting the former star would be pulled in to the black hole. Sense black holes do not give off light, it would appear that the planets are going around 'nothing', and slowly getting closer to that 'nothing'.

The gravity of a black hole is so immense it sucks dust molecules and other materials from nearby stars and dust clouds. As these dust particles accelerate and heat up, they emit x-rays. Objects that emit x-rays can be found by x-ray telescopes.

The third and final way that black holes can be detected is through a technique called gravity lensing. Gravity lensing is when a black hole gets in between a star and the earth. "The black hole is like a lens that bends the star's light rays and focuses them on the Earth". (www.rdrop.com/users/green/school/detect.htm) If you were standing on the Earth looking at the star it would appear to brighten. "The general relativity theory suggests that light should follow the path of time and space which is bent by the black hole's gravity". (www.rdrop.com/users/green/school/detect.htm) This is the method we are trying to simulate.

## Method of Solution

Our method of simulating the detection of black holes is a replication of a method given by Joshua Barnes (http://www.ifa.hawaii.edu/faculty/barnes/barnes.html).

We used the method of detecting black holes using gravity lensing. We placed the black hole between the Earth and a constellation of stars and gave a before and after picture of what the constellation would look like from Earth's surface.

First, we gathered data on the position of stars as seen from Earth, before the black hole is placed between the Earth and that constellation. We selected the Pleiades star cluster for our background on which to do our gravitational manipulation.

Next, we assigned each star a number and made a list of all of the stars we graphed. We wrote down their names, as well as their positions using the X and Y coordinate system on the graph paper. After graphing the stars, we measured how wide our map was, based off of the graphing, in order to determine the map's boundaries.

Then, we had to program a representation of this map using the coordinates of the stars in a structure array. We conducted testing to make sure that the program worked correctly with the original constants given in our example.

The next step was to place the black hole on the map and move each star accordingly. To do this, we determined the mass of the black hole and the distance of the hole from Earth from our example. We used the example's mass of the black hole, 1000000 * the mass of the sun, and its distance from the Earth to the black hole which equaled 4,000 AUs in our original program. Then, we added the steps to the program necessary to output the original X and Y coordinates for each star, and then the new X and Y coordinates after the black hole's gravitational manipulation. In

our original program, we placed the black hole at (26, 30) on the map. We then re-plotted each star in that program by using the Einstein Angle equation to determine their new positions. Then we outputted this to the screen in our original program and printed out charts of these star maps, which is shown in Appendix B. . Then, after we got the program to work with the constants given by Joshua Barnes, we changed the program so that it is user friendly. Then we changed certain constants to variables so that the user can change certain aspects of our program. We made it able to take in values for the placement of the black hole in (X, Y) values, the mass of the black hole in kilograms and the distance of the black hole from the Earth in AUs.

## Conclusion

Among the many conclusions reached in our project, these are a few: If we double the mass of the black hole we get a larger Einstein Angle and the stars move over a million coordinates on the X and Y axis. The smaller we make the distance from earth and the mass of the black hole the smaller the visible change of the position of the stars. To see any movement you have to look at $10^{th}$ to $20^{th}$ decimal point.

If you place the black hole directly on a star, the black hole covers the star and makes it not visible, so the program outputs that the star's coordinates are not a 'number'. We think this is because the light is bent so much you couldn't see the light at all. If you place the black hole near a star it seems to have a greater effect on the star, we think this might be because it has a greater gravitational deflection on the light of the star. The theory of super massive black holes being at the center of our galaxy is clear proven false by our project because we wouldn't see the center of the galaxy we would see a massive "O" of stars surrounding a black area.

## Acknowledgements

We would like to thank all of our sources and mentors, including Joshua Barnes, Ms. Avery, Mark O'Brian, John Chang, Jens Madsen Jr., and Dr. Allen. We are thanking Joshua Barnes for providing the basic equations and background information that we needed to get started on our project. We are thanking Jens Madsen Jr. for helping with the Einstein angle and giving us good ideas to use in our program, report and presentation. We are thanking Dr. Allen for talking to us about math modeling and how to do it correctly.

## References

Chang, John. Black Holes. Available http:// www.eclipse.net/~cmmiller/BH/blkbh.html

September 30, 1998.

Miller, Chris. Black Hole Detection. Available

http://www.rdrop.com/users/green/school/detect.htm

http://www.blackholes.com

http://www.space.com/scienceastronomy/headlines-4.html

Joshua Barnes: http://www.ifa.hawaii.edu/~barnes/lensing/lensing.html

## Appendix A

## Code

```
/*
  Name: Star Light Deflection as seen on Earth caused by a black hole
  Author: Eric Geib, Jens Madsen, Eric Owen, Jason Gunn and Brian Masters.
  Description: simulation of the detection of a black hole
  Date: 03-05-02
  Copyright: (C) 2002
  This program was started on 01-22-02
*/

#include <iomanip.h>
#include <iostream.h>
#include <stdlib.h>
#include<math.h>

int main()                              // begin main function
{
long double R;                                  // distance between the Black hole and the
original position of the star.
```

```
long double R1;                                  // total distance from Black hole to the new
position of the star.
long double D;                                  // Distance to black hole in AUs.
long double AC;                                  // Einstein angle variable
const long double C = 299792458.0;                  // speed of light Meters / second
const long double G = 6.67259;                  // Gravitational constant
const long double PI = 3.14159265358979323846264338327950;    // pi
long double RC;                                  // radius of the circle created by the black hole's
gravity
float L = 30.0;                                 // One degree on our graph paper
long double X_SQUARED;                              // These variables are the numbers that
are getting squared
long double Y_SQUARED;                              // These variables are the numbers that
are getting squared
long double STARXX;                                 // X variable used for pow equations
long double STARYY;                                 // Y variable used for pow equations
float X_BH;                                     // variable for user to input the X coordinate of
the black hole
float Y_BH;                                     // variable for user to input the Y coordinate of
the black hole
long double D1;                                 // Variable for user to input AU's.
long double MBH;                                // Variable for the user to input the black hole's
mass
```

```
struct STAR                                  // structure for star coordinates
{
double X;
double Y;
};



STAR STAROUT[47]= {2,20,6,21,7,19,9,12,9,35,10,46,11,56,8,59,20,25,24,21,26,20,
28,17,26,23,27,28,29,34,30,37,32,37,35,36,34,34,32,32,37,34,36,33,36,30,37,29,
32,28,33,25,34,22,33,20,36,18,30,22,29,23,41,36,43,20,44,41,42,50,49,57,48,51,
45,15,30,28,46,13,52,19,53,8,52,44,54,41,56,41,27,32,30,30};
// starout is the structure that holds the 47 original star positions


STAR STARNEW[24];
//starnew is the structure that holds the 47 new star positions
do  // The loop for input of the X coordinate of the black hole
  {
   cout <<"Where do you want the black hole to be on the X axis" << endl;
   cout <<"(Please pick a number between 1 and 60):" <<endl;
   cin >>X_BH;
  }
  while (X_BH < 1 || X_BH > 60);
```

```
cout << "" << endl;

do // The loop for input of the Y coordinate of the black hole
 {
  cout <<"Where do you want the black hole to be on the Y axis" << endl;
  cout <<"(Please pick a number between 1 and 70):" <<endl;
  cin >>Y_BH;
 }
  while (Y_BH < 1 || Y_BH > 70);

cout << "" << endl;

do  // The loop for input of the AU's
 {
  cout <<"What do you want to set the distance to the black hole from the Earth to be"<<endl;
  cout <<"(this measurement is in AUs, do not pick negative numbers please):"<<endl;
  cin >>D1;
 }
 while (D1 <= 0);

cout << "" << endl;
```

```
do  // The loop for input of the mass of the black hole
 {
  cout <<"What do you want to set the black hole's mass equal to"<<endl;
  cout <<"(this measurement is in kilograms, do not pick negative numbers please):"<<endl;
  cin >>MBH;
 }
 while (MBH <= 0);

cout << "" << endl;

for (int q = 0; q < 47; q++)      // loop for output of stars
{
X_BH;                                    // X coordinate of the black hole
Y_BH;                                    // Y coordinate of the black hole

long double AC2;                              // Variable for the Einstein Angle
long double AC3;                              // Variable for the Einstein Angle
const long double MS = 20000000000000000000.0;          // mass of the sun in kilograms
long double M = MBH * MS;                          // mass of the black hole in kilograms
long double AU = 149604970;                       // Distance to the Earth in kilometers
D = D1 * AU;                                  // equal to 598419880000 kilometers
```

**AC = (M/MS)/(D/AU); // first part of Einstein Angle equation, using mass of sun, the mass of the black hole, the distance to the blackhole and the AU units in kilometers**

**AC2 = sqrt(AC); // second part of the Einstein Angle equation, does the square rooting part of the equation**

**AC3 = AC2*0.0115; // last part of the Einstein Angle equation, does the last steps to find the answer : .18 degrees**

**cout << "" << endl;**

**cout << "If you happen to place the black hole on a star, it will appear as -NaN instead of it's new coordinates." << endl;**

**cout << "" << endl;**

**cout << setprecision(5)<< AC3 <<" is the Einstein Angle "<< endl;**

**cout << "" << endl;**

**RC = AC3 * L; // Equation for the radius of the circle created by the black hole's gravity**

**X_SQUARED = X_BH-STAROUT[q].X; // Equation used to find the X coordinate number that is to be squared for the distance from the black hole equation**

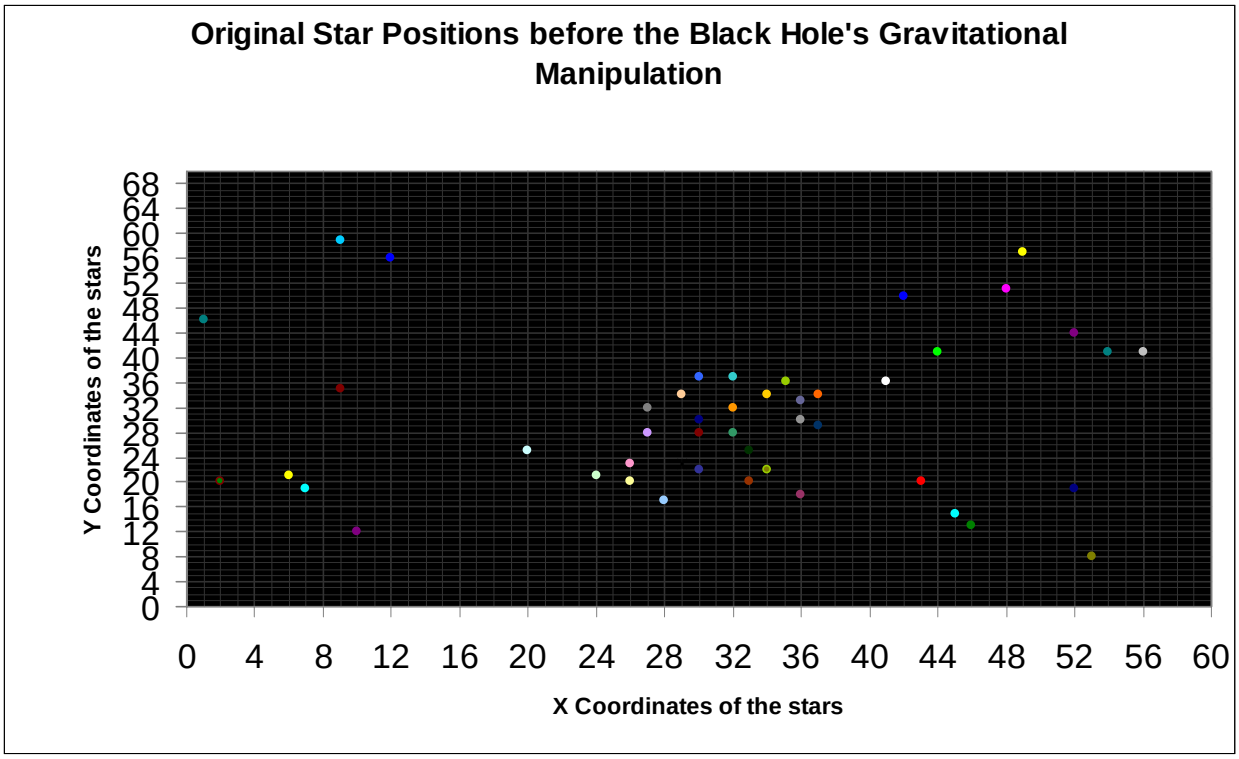
**Y_SQUARED = Y_BH-STAROUT[q].Y; // Equation used to find the Y coordinate number that is to be squared for the distance from the black hole equation**

```
STARXX= pow(X_SQUARED,2.0);                    // Equation used to square the X
coordinate; this number will be used in the distance from the black hole equation
STARYY= pow(Y_SQUARED,2.0);                    // Equation used to square the Y
coordinate; this number will be used in the distance from the black hole equation

R = sqrt(STARXX+STARYY);                       // Equation used to find the distance
from the black hole to the original position of the star

R1 = R+((RC*RC)/R);                            // Equation used to find the distance from
the black hole to the new position of the star

STARNEW[q].X = X_BH + ((R1/R)*((STAROUT[q].X) - X_BH));      // Used to find the
new X coordinate of the star after gravitational deflection
STARNEW[q].Y = Y_BH + ((R1/R)*((STAROUT[q].Y) - Y_BH));      // Used to find the
new Y coordinate of the star after gravitational deflection
cout.setf(ios::fixed);
cout <<setprecision(0)<<STAROUT[q].X<<" is the old X coordinate for star
"<<q+1<<'\n';     // Output of the X coordinate of the original position of the star
cout <<setprecision(2)<< STARNEW[q].X<<" is the new X coordinate for star " <<q+1<<
'\n';   // Output of the X coordinate of the new position of the star

cout << "" << endl;
```

```
cout <<setprecision(0)<<STAROUT[q].Y<<" is the old Y coordinate for star "<<q+1<<endl;      // Output of the Y coordinate of the original position of the star
cout <<setprecision(2)<< STARNEW[q].Y<<" is the new Y coordinate for star " <<q+1<< '\n';   // Output of the Y coordinate of the new position of the star

system("PAUSE");
system("cls");
}
return 0;
}
```

## Appendix B

### Original Star Chart



# Appendix C

## Original Star Chart