Addendum to:
Image Comparisons of Black Hole vs. Neutron Dark Star Ray Tracing

D.T. Froedge

Formerly Auburn University
Phys-dtfroedge@glasgow-ky.com

V110116
@ http://www.arxdtf.org

ABSTRACT

In the previous paper computer programs were developed to illustrate the image comparison between a conservative local energy theory gravitational theory and a black hole. A local conservative theory of gravitation[1] does not generate a Black Hole, but a very heavy neutron star (NDS) having a radius only slightly larger than the gravitational radius [2],[3],[4],. The light propagation formula used in the original paper was the standard symmetric Shapiro propagation formula for flat space, well tested as the proper projection of the GR metric onto a flat space, for star grazing and Shapiro delay. Unfortunately due to the asymmetric nature of the propagation of light related, this expression does not properly represent photon trajectories near the gravitational radius. This paper redoes those projections and creates a star image consistent with the proper index of refraction for photons near a massive star.

Introduction

By projecting the photon trajectories of the Einstein metric onto flat space Blandford & Thorne [5] have shown the tangential index of refraction of light induced by gravitating body is:

$$\eta_\theta = \left(1 - 2\frac{\mu}{r}\right)^{-1} \tag{1}$$

Karimi, & Khorasani [6], have shown that with a more detailed development of the asymmetric aspects of the GR metric, that the index of refraction is more accurately represented as:

$$\eta = (1+\phi)^{-1/2}\left(1+\phi\cos^2\theta\right)^{1/2} \quad , \qquad \phi = \frac{r_s}{r} = \frac{2\mu}{r} \tag{0.2}$$

The angel θ is the angle between the wave vector and the radius. By dropping second order terms and simplifying, this can be written as:

$$c = \left(1-\left(1+\cos^2\psi\right)\frac{\mu}{r}\right) \tag{0.3}$$

The Schwarzschild radius of the gravitational potential in GR results from energy transfers to the field which creates an event horizon. For a local energy conservation theory in which a photon exchanges no energy with the field, the barrier at the Schwarzschild radius does not exist and thus the expression must be the quadratic alternative:

$$\left(1-2\frac{\mu}{r}\right) \rightarrow \left(1-\frac{\mu}{r}\right)^2, \tag{0.4}$$

For such a theory the Karimi expression Eq.(0.2), must also be modified to the quadratic version:

$$c = \left(1-\frac{\mu}{r}\right)^{\left(1+\cos^2\theta\right)} \tag{0.5}$$

The change from Eq.(0.3), to Eq.(0.5), has little effect on physical phenomena accept in the local proximity of a Black Hole, and thus for this presentation it is necessarily the proper expression.

The changes to the trajectories of particles leaving the surface of a heavy neutron star are significantly different from the earlier paper. With this expression for the index of refraction there are no trapped or orbital trajectories that preclude a photon from the surface, escaping into space.

Photons leaving the surface from the side opposite the observer wrap and focus in a direction along the line of sight giving the appearance of a bright ring surrounding the star. The brightness increases toward the edge which has a diameter of about 10.6μ. This is almost exactly the same as the diameter of the shadow of the black hole.

This size conscience makes the observational distinction between the black hole and the NDS less discernable than for the non-asymmetric propagation. The distinction however comes from the fact the radiation from a Black Hole comes entirely from the exterior of the shadow or circle, and radiation from the NDS comes entirely from the interior of the circle. Unless the star has a protruding jet, the outer edge of the NDS is quite sharp, thus the EHT observations should be able to discern the difference.

The procedure and computer code used in this paper for defining the image of the heavy neutron star NDS is similar to the program used in the earlier paper and is included in Appendix I. Reference to that paper may be useful.[1]

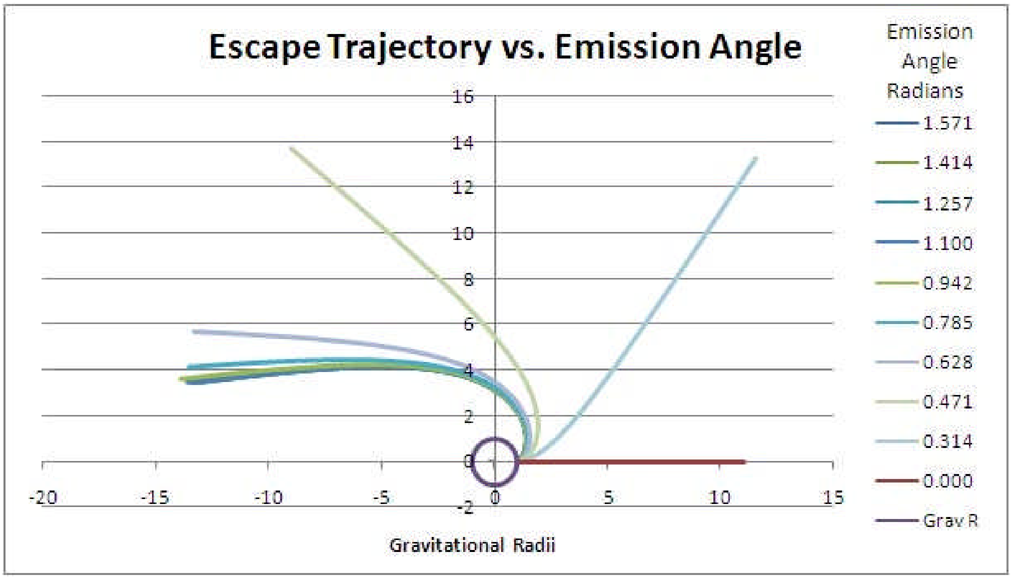
As noted in earlier papers the surface of a star in a conservative gravitational theory the size of Sag A* is at about 1.025 μ[2].

General

The image of the star form an outside observer is created by starting from a series of trajectories at a single point. By changing the emission angle from a vertical, zero to $\pi/2$ the non-linearity of the propagation And the exit angle from that point can be developed.

As the angle increases the trajectories start wrapping around the limb until at the total angle of wrap at $\pi/2$ is about 3.27 radians. As shown in figure 1, at an emission angle greater than about .94 rad the wrap angle is about the same, indicating a focusing of the trajectories in a tight column. In contrast to the earlier paper there are no orbits or non-escaping trajectories.

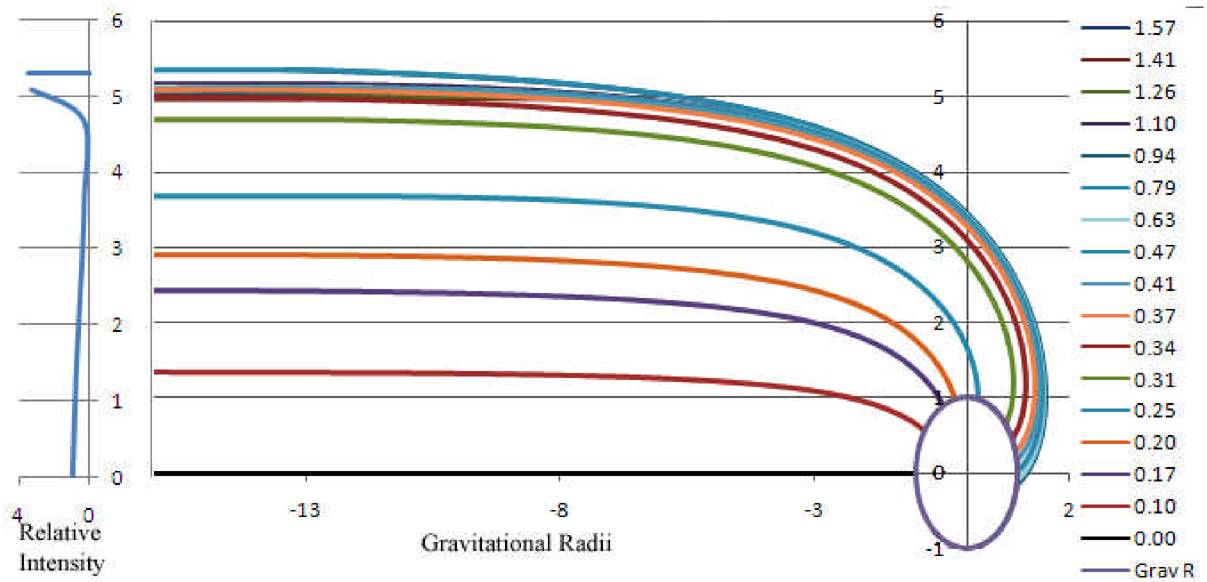
Figure 1



The next step is to take each of the calculated trajectories and rotate the origin on the surface such that the exit direction is parallel to the line of sight for a distant observer along the radius in the π direction.

A rotation of the emission trajectory around the surface to the point that the trajectories are parallel to the line of sight of a distant observer renders the maximum lateral displacement of a photon being emitted from that point. By considering all the trajectories, the maximum lateral displacement for any photon leaving the surface can be determined. Figure 2 shows those trajectories.

Figure 2



From this it is seen that the maximum radial displacement is about 5.3 μ, thus giving a diameter of 10.6 μ. This is nearly the same diameter as the shadow of a Black Hole with the same mass.

Intensity

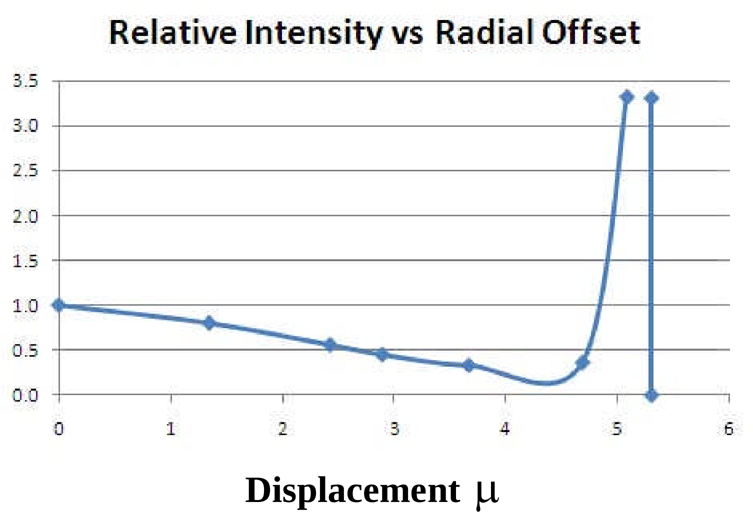
The intensity of a point on the surface as observed at a distance can be determined by calculating the relative angular dispersion of the trajectories when they exit the surface, and when they escape. Taking the ratio of the solid angle between two trajectories displaced by 0.1 rad at the surface and on escape determines the relative intensity of the point as observed by a distant observer. With that, and the horizontal displacement of the trajectory gives the relative intensity of the image.

Table 1

Exit Displac-ement	Photon Emission Angle	Photon Exit Angle	Relative Intensity
5.30	1.571	3.267	1.53E+10
5.12	1.414	3.267	1.31E+08
5.00	1.257	3.266	1.40E+06
5.17	1.100	3.265	9.98E+04
5.30	0.942	3.260	1.25E+04
5.30	0.785	3.247	1.84E+03
5.11	0.628	3.212	2.38E+02
5.08	0.471	3.101	3.31E+00
4.68	0.314	2.442	3.62E-01
3.67	0.250	1.788	3.30E-01
2.89	0.200	1.322	4.50E-01
2.43	0.170	1.078	5.59E-01
1.35	0.100	0.590	8.01E-01
0.00	0.000	0.000	1.00E+00

A plot of the intensity vs the radial offset displacement is shown in Figure 3, and is shown plotted against the trajectories in Figure 2.

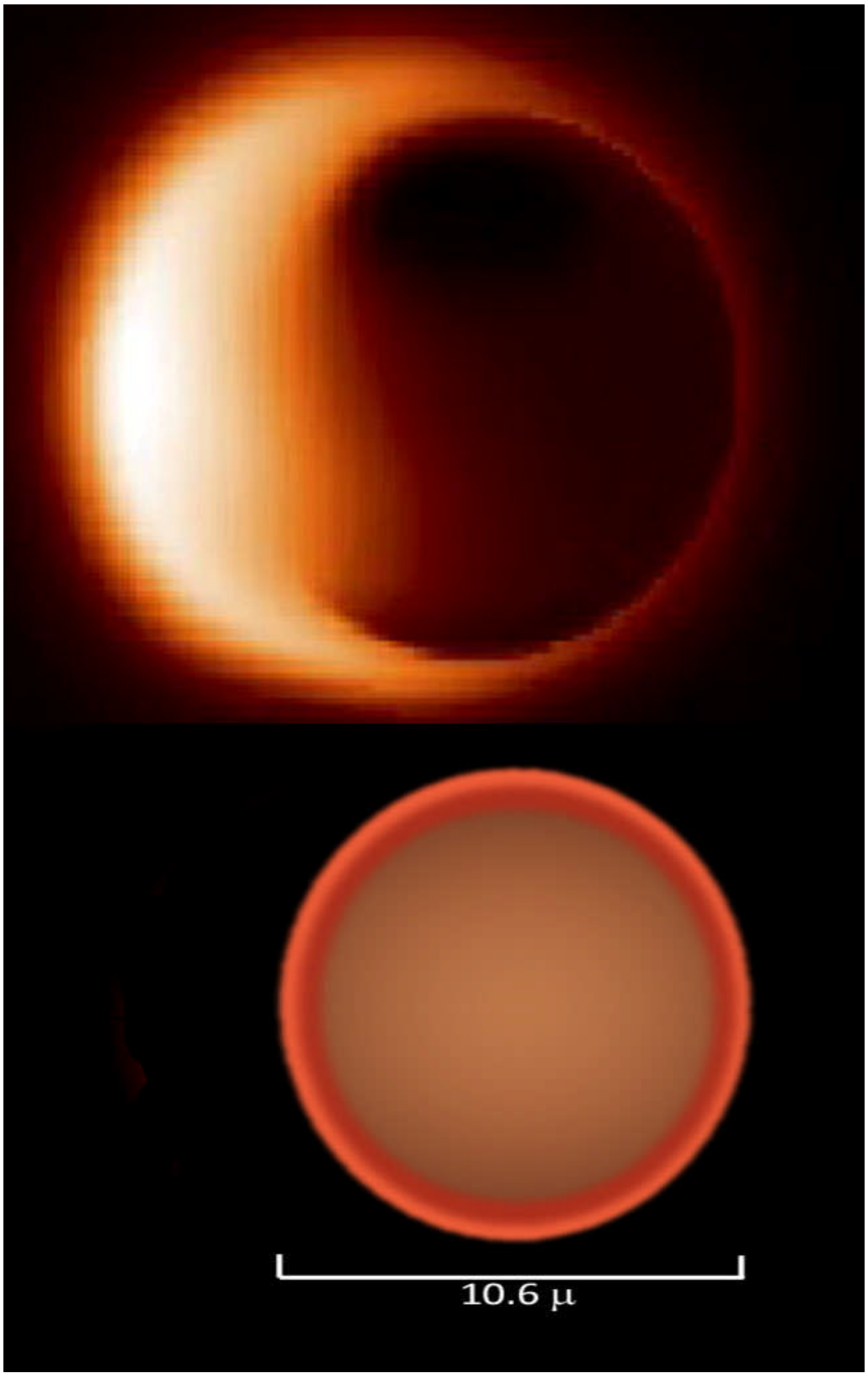
Figure 3



Note that the relative intensity at the edge goes very high in comparison to the intensity of the center. This is due to the lens like focusing of the image from the center of the back side. This focusing will produce an image with a bright ring at a diameter of about 10μ.

Image

From the foregoing calculations, an image can be developed for comparison with the expected Image of a Black Hole produced by the EHT team [7]



Top – Accretion driven radiation image of the shadow of a Black Hole[7]
Bottom - Alternative heavy neutron star image

Conclusion

There is a significant difference between the anticipated Black Hole image of Sag S* as produced by the Event Horizon Telescope (EHT) research team, and the image of a heavy Neutron Dark Star as derived from a local conservation of energy theory of gravitation. The measurements taken in the upcoming year should provide the answer.

References:

1. DT Froedge, Scalar Gravitational Theory with Variable Rest Mass,V020914, http://www.arxdtf.org/

2. DT Froedge, Black Hole vs. Variable Rest Mass Neutron Star, V041912, http://www.arxdtf.org/

3. DT Froedge, Image Comparisons of Black Hole vs. Neutron Dark Star by Ray Tracing V070114, http://www.arxdtf.org

4. DT Froedge, Gravitation is a Gradient in the Velocity of Light V081216, http://www.arxdtf.org/

5. Roger Blandford, Kip S. Thorne, in Applications of Classical Physics, (in preparation, 2004), Chapter 26 http://www.pma.caltech.edu/Courses/ph136/yr2002/chap26/0226.1.pdf

6. F. Karimi, S. Khorasani, Ray-tracing and Interferometry in Schwarzschild Geometry, arXiv:1001.2177 [gr-qc]

7. Reconstruction of Static Black Hole Images Using Simple Geometric Forms Leonid Benkevitch, et al. MIT Haystack Observatory, September 2, 2016 arXiv:1609.00055v1 [astro-ph.IM] 31 Aug 2016

Appendix I

Basic Program used for Projections

```
'         Gravitational Ray Trace
'              DT Froedge
'            copyright 2014
'   GR black hole, and VRM Dark Star
'This vweaion only for anisotropic NDS
'This is a basic program that calculates ray traces
'in the vicinity of black holes and varible mass stars
'in double precision.
'Output is a comma delimited text file that can be imported into Excel
'or other spreadsheets for graphing.
'Program calculates in the first quadrant (quad)by rotation
'then corrects output to proper quadrant by reversing this.
'program only handles counter clockwise motion.
'distance is in gravitationsl radii. c=1, Mu = 1
'Try to do a smother transform that covres all quadrants

'This the clean up of the first working asymetric propagation for the nds
' v = (1 - / r#(2)) ^ (1 - (COS(Radtheta#)) / 2)
'it was done by calculating the value of the angle change as a result of the change in
the
'radius and then calculating the velocity as a result of a change in the angle

'                User choice of propagation formula
'fprmat%=1 then v#(1) is (1 - 1 / r#(1)) ^ 2   Dark Star. Original non asymetric
propagation
'fprmat%=2 then v#(1) is (1 - 2 / r#(1))       GR Black Hole Shapiro velocity
'fprmat% = 6     'v = (1 - / r#(2)) ^ (1 - (COS(Radtheta#)) / 2)

CLS
'Sag a*        mass 4.1 e6   0.6 e6 suns           =  8.154572e39  grams
   'Gm/c^2=     Mu = 605181439025  cm     =  20.186  light seconds

pi# = 3.14159265358979#
quad% = 1                        'start quadrant
Delt# = .0001                    'increments of time

x1# = 1.025    '3.7387708             'initial x position  (to be set by user)
y1# = 0                               'initial y position  (to be set by user)
P1# = .47#                            'initial angle       (to be set by user)
r0# = (xo# ^ 2 + yo# ^ 2) ^ .5   'initial radius       defined


seq% = 0                     'initial sequencer on =1
nonstop% = seq%
delp# = .008                 'step size angle
nseq% = 20                       'number of sequential steps

xxo# = x1#
yyo# = y1#
po# = P1#
EQN% = fprmat%
fun% = 30

NEXin:
st% = 2
fun% = fun% + 1
GOSUB sequence

GOSUB Datain                 'alows inputs initial conditions
CLS
```

```
x1# = xxo#
y1# = yyo#
P1# = po#
fprmat% = EQN%



Formprt$ = " (1 - 1/ r#(2)) ^ (1 + (COS(Radtheta#))^2 "   'only propagation index used
in this version program
                  ' Not used this version
'IF fprmat% = 2 THEN Formprt$ = "shapiro v = (1 - 2 / r#) "
'IF fprmat% = 3 THEN Formprt$ = "Karimi Index Eq 5b"
'IF fprmat% = 4 THEN Formprt$ = "Anisotropi Karimi, Khorasani  Eq 13 "
'IF fprmat% = 5 THEN Formprt$ = "Component Karimi, Khorasani  Eq.14"


OutFile$ = "OutD" + STR$(fun%) + ".txt"
'OutFile$ = "Outall.txt"          ' file out
OPEN OutFile$ FOR OUTPUT AS #2

  ' Output format
'PRINT #2, " x --------- y ---------  rx----- ry----- r------ dir---------    Theta1--
----    theta2------    delt--------    AngPos------    deltv-----  dist---  quad--
seconds";

PRINT #2, "x,y,r,dir,vrad,vthet,vel,velAngTh, ang
pos,gravX,gravY,refa#,steps,seconds";
PRINT #2, ",";
PRINT #2, Formprt$
  'Initial set
grad# = 1                           'defines Circle at gravitational radius
p#(1) = P1#                         'initial velocity direction angle from 1 to 2
x#(1) = x1#                          'initial first point x position
y#(1) = y1#                          'initial first point  y position
r#(1) = (x#(1) ^ 2 + y#(1) ^ 2) ^ .5  'first point r & v values
   'velocity at point1
ap# = ATN(y1# / x1#)       'watch
xp# = x#(1): yp# = y#(1)    'CALCULATE angular position of particle 1 Ap# with Xp# and
Yp#
GOSUB angularPos             'gets ap#  and tangent angle Radtheta# = P#(1) - ap#

v#(1) = (1 - (1 - (COS(Radtheta#) / 2)) / r#(1)) ^ 2



   'initial print
  x#(2) = x#(1)
  y#(2) = y#(1)
  r#(2) = r#(1)
  p#(2) = p#(1)
  GOSUB procede


   ' calculate 2nd point  based on initial conditions


vx#(1) = v#(1) * COS(p#(1))                         'initial velocity components
vy#(1) = v#(1) * SIN(p#(1))
x#(2) = x#(1) + vx#(1) * Delt#                      '2nd point location running along p
y#(2) = y#(1) + vy#(1) * Delt#        '

here:                                               'Recycle for next point

xp# = x#(2): yp# = y#(2)                             'CALCULATE angular position ap#
with Xp# and Yp# of particle 2
r#(2) = (x#(2) ^ 2 + y#(2) ^ 2) ^ .5                 'r  second point
```

```
GOSUB angularPos                                   'and tangent angle Radtheta# =
P#(1) - ap# , P#(1)  is initial direction of particle 2


  IF y#(2) > 0 THEN                               ' 1st  second quad
    'angular position
 IF x#(2) = 0 THEN a#(2) = pi# / 2: GOTO skip

 IF (y#(2) / x#(2)) > 0 THEN       '1st quad
 IF (y#(2) / x#(2)) <= 10 THEN a#(2) = ATN(y#(2) / x#(2)): GOTO skip
 IF (y#(2) / x#(2)) > 10 THEN a#(2) = pi# / 2 - ATN(x#(2) / y#(2)): GOTO skip
 END IF

 IF (y#(2) / x#(2)) < 0 THEN                  'second quad
  IF (-y#(2) / x#(2)) >= 10 THEN a#(2) = pi# / 2 + ATN(-x#(2) / y#(2)): GOTO skip
  IF (-y#(2) / x#(2)) < 10 THEN a#(2) = pi# - ATN(-y#(2) / x#(2)): GOTO skip
 END IF

  END IF

skip:

 IF y#(2) < 0 THEN                 'tigger point 'third quadrant reset to first quad.
Not
  'working and no need since photon dosen't orbit
   st% = st% + 1
   IF st% = 4 THEN
  st% = 3: a#(2) = a#(2) - pi#
  GOSUB procede
  GOTO Plast                 'one rotation stopstop
 END IF

  x#(2) = -x#(2)                 'resign 3rd point to second and 2nd  point to 1st,
recycle
  y#(2) = -y#(2)
  p#(2) = p#(2) - pi#            'resets direction
  vx#(1) = -vx#(1)
  vy#(1) = -vy#(1)

END IF

vx# = vx#(1)                                           'initial component velocity
from 1 arriving at second point (3rd pt)
vy# = vy#(1)

GOSUB calcradv                                         'rotates position and
velocities to angular position theta=0
  'to falilitate snells law calculations
 ' Apply snell angle change to radial cordinate velocities
v#(1) = (1 - (1 - (COS(Radtheta#)) / 2) / r#(1)) ^ 2
v#(2) = (1 - (1 - (COS(Radtheta#)) / 2) / r#(2)) ^ 2


rep:                                        'snell cond first change in radial
position and then
    'change invelocity due to asymetry of propagation
n = n + 1
xx# = ((v#(2) / v#(1)) * SIN(Radtheta#))   ' visual basics arcsin function
IF xx# > 1 THEN xx# = 1 - .00000001#
yy# = (Radtheta# - pi# / 2)                'visual basics arccos function


'PRINT USING "+##.########"; Radtheta#;    'testing
'PRINT USING "+##.########"; vrad#(2);
IF Radtheta# > pi# THEN Radtheta# = pi# / 2 + ATN((-yy# / SQR(-yy# ^ 2 + 1))): GOTO
skkip3  'axis issues
IF Radtheta# <= pi# THEN Radtheta# = ATN(xx# / SQR(-xx# * xx# + 1))
```

```
skkip3:

v#(2) = (1 - (1 - (COS(Radtheta#)) / 2) / r#(2)) ^ 2

'delRad# = Holdrad# - Radtheta#      'testing functions
'Holdrad# = Radtheta#                'can report change in direction due to asymetry

IF n = 1 THEN GOTO rep ELSE n = 0      'Asymetry contribution

 vtheta#(2) = v#(2) * SIN(Radtheta#)
 'vrad#(2) = v#(2) * COS(Radtheta#)             'alternate calculation- same value
  vrad#(2) = (v#(2) ^ 2 - vtheta#(2) ^ 2) ^ .5



'PRINT USING "+##.########"; vrad#(2)          'Test reports
'PRINT USING "+##.########"; Radtheta# ;
'PRINT USING "+##.########"; Radtheta# ;

IF vrad#(2) = 0 THEN refa# = pi# / 2: GOTO skip3
refa# = ATN(vtheta#(2) / vrad#(2))

skip3:

'IF vrad#(2) < 0 THEN                  'Test stops
'  STOP
'END IF


'PRINT USING "+##.########"; v#(1);      'Test reports
'PRINT USING "+##.########"; v#(2);
'PRINT USING "+##.########"; r#(1);
'PRINT USING "+##.########"; r#(2)
  ' PRINT
  ' PRINT



GOSUB calccartv                                                    'rotates cord to
lab velocities to back 'vx#(2) vy#(2)

   'calculate direction
 'vy = actual y velocity                                            ' departing cart
cord    vx#(2) vy#(2)
   'vx = actual x velocity
  x#(3) = x#(2) + vx#(2) * Delt#                                    'third point
positiont  vx#(2) vy#(2)
  y#(3) = y#(2) + vy#(2) * Delt#                                    'y vy t
  r#(3) = (x#(3) ^ 2 + y#(3) ^ 2) ^ .5                              'r 3rd point

 IF vy#(2) >= 0 THEN
'1st and 2nd quad
IF vx#(2) > 0 THEN                           'first
'1st quad
IF vy#(2) = 0 THEN p#(2) = 0: GOTO skip4
'0
IF (vy#(2) / vx#(2)) < 10 THEN p#(2) = ATN(vy#(2) / vx#(2)): GOTO skip4
'90 degree prob
IF (vy#(2) / vx#(2)) >= 10 THEN p#(2) = pi# / 2 - ATN(vx#(2) / vy#(2)): GOTO skip4
END IF
IF vx#(2) <= 0 THEN
'second quad
  IF vx#(2) = 0 THEN p#(2) = pi# / 2: GOTO skip4
  IF (-vx#(2) / vy#(2)) > 10 THEN
 p#(2) = pi# - ATN(-vy#(2) / vx#(2)): GOTO skip4
  END IF
```

```
    IF (-vx#(2) / vy#(2)) < 10 THEN p#(2) = pi# / 2 + ATN(-vx#(2) / vy#(2)): GOTO skip4
 END IF

END IF

IF vy#(2) <= 0 THEN                                     'P in 2nd quad
   IF vy#(2) = 0 THEN p#(2) = pi#: GOTO skip4
   IF vx#(2) <= 0 THEN
     'top
    IF (vx#(2) / vy#(2)) > 10 THEN p#(2) = pi# + ATN(vy#(2) / vx#(2)): GOTO skip4
     'bottom position near pi
    p#(2) = 3 * pi# / 2 - ATN(vx#(2) / vy#(2)): GOTO skip4
   END IF


END IF

   'reasignments of particle 3 to 2 and 2 to 1

skip4:


     'leaving direction angle from 2


  x#(2) = x#(3)               'resign first point, for recycle
  y#(2) = y#(3)
  r#(1) = r#(2)
  r#(2) = r#(3)
  p#(1) = p#(2)               'depature dir at p 2 arrival direction at 3rd point
  vx#(1) = vx#(2)             'component velocity leaving point 2 arrivine point3
  vy#(1) = vy#(2)
  v#(4) = v#(1)               'v#(1)     hold for print
  v#(1) = v#(2)

GOSUB PUTITOUT5           'data out


GOTO here                 'loop back

    'Subs

calcradv:            'Rotates velocity and particle cords to particle position to
  'zero theta to facilitate angle change in velocity

   aa#(2) = a#(2)
   IF aa#(2) > pi# THEN aa#(2) = aa#(2) - pi#     ' cycling, dosent happen in this
version
   vrad# = vx# * COS(aa#(2)) + vy# * SIN(aa#(2))
   vtheta# = vy# * COS(aa#(2)) - vx# * SIN(aa#(2))

RETURN

calccartv:           ' Rotates cordinates back to original positionss after snell
angle change

   aa#(2) = a#(2)
   IF aa#(2) > pi# THEN aa#(2) = aa#(2) - pi#     ' cycling
   vx#(2) = vrad#(2) * COS(aa#(2)) - vtheta#(2) * SIN(aa#(2))
   vy#(2) = vrad#(2) * SIN(aa#(2)) + vtheta#(2) * COS(aa#(2))

RETURN

angularPos:               'CALCULATE angular position Ap# with Xp# and Yp#
   ' and define angle between velocity and radius vector

 IF xp# = 0 THEN ap# = pi# / 2: GOTO skipit
```

```
 IF (yp# / xp#) > 0 THEN
 IF (yp# / xp#) > 10 THEN ap# = pi# / 2 - ATN(xp# / yp#): GOTO skipit
 ap# = ATN(yp# / xp#): GOTO skipit
 END IF

 IF (yp# / xp#) < 0 THEN                  'second quad
  IF (-yp# / xp#) > 10 THEN ap# = pi# / 2 + ATN(-xp# / yp#): GOTO skipit
ap# = pi# - ATN(-yp# / xp#): GOTO skipit
 END IF
 IF ABS(yp# / xp#) < .00000005# THEN     'rountoff prob             'second quad
ap# = 0
 END IF

skipit:

   Radtheta# = p#(1) - ap#     'asymetry angle to calculate velocity change
  'or tangential velocity angle betweem velocity
  'and radial direction or snell incident angle.

RETURN


sequence:                      'This routine gives sequential files with changes in
initial direction
 IF seq% = 1 THEN
  po# = po# + delp#
  seq2% = seq2% + 1
  IF seq2% > nseq% THEN seq2% = 0: seq% = 0: CLOSE : END
 END IF
RETURN



PUTITOUT5:                                      ' outputs location outfile
sht% = sht% + 1                            'interation counter
distance# = distance# + v#(1) * Delt#

  'Stop conditions
IF r#(2) <= 1 THEN GOTO Plast
IF x#(2) > 14 THEN GOTO Plast
IF y#(2) > 14 THEN GOTO Plast
IF x#(2) < -14 THEN GOTO Plast
IF y#(2) < -14 THEN GOTO Plast

'IF r#(2) > 1.03 THEN GOTO Plast  'temp

K$ = ""
K$ = INKEY$: IF K$ <> "" THEN CLOSE : STOP
IF sht% = 10000 THEN sht% = 0: sht2% = sht2% + 1: GOTO procede  'output one out of
10000

  'IF sht% > 1000 THEN STOP                 ' test conditions to find and print
problems
  'IF sht% > -1 THEN GOTO procede
  'IF sht2% = 8 AND sht% = 6895 THEN STOP
  'IF  sht2% = 11 and sht%> 6500 then
  'IF  sht2% = 0 and sht% < 500 then  GOTO procede

RETURN

procede:
'IF sht2% = 140 THEN GOTO Plast                      'Stop conditions
Rx#(2) = grad# * COS(sht2% * pi# / 20)    'gravitational radius
Ry#(2) = grad# * SIN(sht2% * pi# / 20)
```

```
'PRINT #2, USING "+##.###########"; x#(2) * ((-1) ^ st%);             'x
'PRINT #2, ",";
'PRINT #2, USING "+##.###########"; y#(2) * ((-1) ^ st%);              'y
'PRINT #2, ",";
PRINT USING "+##.###########"; x#(2) * ((-1) ^ st%);              'x   window printing
PRINT USING "+##.###########"; y#(2) * ((-1) ^ st%);              'y
PRINT USING "+##.###########"; r#(2);                             'r
PRINT USING "+###.##########"; p#(2) + pi# * (st% - 2)          'rotates back to
proper quad

   'File prints

PRINT #2, USING "+##.###########"; x#(2) * ((-1) ^ st%);             'x
PRINT #2, ",";
PRINT #2, USING "+##.###########"; y#(2) * ((-1) ^ st%);                      'y
PRINT #2, ",";
PRINT #2, USING "+###.###########"; (x#(2) ^ 2 + y#(2) ^ 2) ^ .5;          'r
PRINT #2, ",";
PRINT #2, USING "+###.##########"; p#(2) + pi# * (st% - 2);
'directiom
PRINT #2, ",";
PRINT #2, USING "+###.#########"; vrad#(2);                               'rad vel
PRINT #2, ",";
PRINT #2, USING "+###.#########"; vtheta#(2);                             ' ang vel
PRINT #2, ",";
  PRINT #2, USING "+###.#########"; (vrad#(2) ^ 2 + vtheta#(2) ^ 2) ^ .5;   'v#(2)
' veloc
PRINT #2, ",";
PRINT #2, USING "+###.#########"; Radtheta#;                                'angular
Radtheta# = angle=0 direction along radius
PRINT #2, ",";
PRINT #2, USING "+###.##############"; (a#(2) + pi# * (st% - 2));          'angular
position
PRINT #2, ",";
PRINT #2, USING "+######.####"; Rx#(2);                                     'x
coordinates of gravitational radius
PRINT #2, ",";
PRINT #2, USING "+###.#########"; Ry#(2);                                   'y
coordinates of gravitational radius
PRINT #2, ",";
PRINT #2, USING "+##.########"; refa#;                                      'Snell
law refraction angle ref ang
PRINT #2, ",";
  sht = sht%
  sht2 = sht2%
PRINT #2, USING "+##########."; sht + sht2 * 10000;                         'steps
PRINT #2, ",";
PRINT #2, sht2% * 20.186;                                                   'seconds
PRINT #2, ","

RETURN



selectN:                        'Index of refraction formulas not used in this
vession

 ' NDS                          b =  3.7387708  >> sha 5.84    orbit3.736    escape
angle .0025
IF fprmat% = 1 THEN
   v# = (1 - 1 / r#) ^ 2
END IF



END IF
                         'only index used in this versiom
```

```
IF fprmat% = 6 THEN
   vtheta1# = (1 - 1 / r#(2)) ^ (1 + COS(Radtheta#) ^ 2)  'at next position
   v# = vtheta1#
END IF

RETURN

resetvar:                     'Reset all variables for next file run
sht% = 0
sht2% = 0
Radtheta# = 0
delvthet# = 0
holdvt# = 0
delvthet# = 0
vtheta#(2) = 0
vrad#(2) = 0
vtheta# = 0
vrad#(2) = 0
refa# = 0
delvthetx# = 0
delvthety# = 0
n = 0: m = 0: vtheta1#(1) = 0: vtheta1#(2) = 0
vtheta1#(3) = 0: vtheta1#(4) = 0: vtheta1#(5) = 0
x#(2) = 0: x#(3) = 0
y#(2) = 0: y#(3) = 0
r#(2) = 0:   r#(3) = 0
p#(1) = 0: p#(2) = 0
vx#(1) = 0: vx#(2) = 0
vy#(1) = 0:   vy#(2) = 0
v#(4) = 0
v#(1) = 0
v#(2) = 0

RETURN
Plast:
CLOSE
IF nonstop% = 1 GOTO NEXin

hold:
   AX$ = ""
  AX$ = INKEY$
  IF AX$ = "" THEN GOTO hold
GOTO NEXin



'XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
Datain:


CLS

 FOR I% = 8 TO 21
 LOCATE I%, 16, 0: PRINT SPACE$(29);
 NEXT I%
LOCATE 7, 19: PRINT "* Graviataional Ray Trace *";
LOCATE 10, 25, 0: PRINT "Initial x value    ";
LOCATE 12, 25, 0: PRINT "Initial y value    ";
LOCATE 14, 25, 0: PRINT "Initial dir, radians";
LOCATE 16, 25, 0: PRINT "Selected Index Eq  ";
LOCATE 18, 25, 0: PRINT "Initial file number";


 LOCATE 25, 5
 PRINT "<Up><Dn> Move      <P> procede        <ENT> Select       <ESC>  Quit ";

GOSUB refresh7:
```

```
'**** select function
IF Retflag% = 1 THEN GOSUB 23760: Retflag% = 0: GOTO 23520
23510 rwfunc% = 10
  GOSUB 23760 '** default
23520 BB$ = "": BB$ = INKEY$
 IF nonstop% = 1 THEN GOTO endin
 IF BB$ = "p" THEN CLS : BB$ = "": GOTO endin                 'exit
 IF BB$ = "P" THEN CLS : BB$ = "": GOTO endin

 IF BB$ = CHR$(27) THEN BB$ = "": CLS : CLOSE : STOP
 IF LEN(BB$) = 2 AND MID$(BB$, 2, 1) = CHR$(72) THEN GOSUB 23800: GOTO 23520
 IF LEN(BB$) = 2 AND MID$(BB$, 2, 1) = CHR$(80) THEN GOSUB 23700: GOTO 23520

 IF BB$ = CHR$(13) AND rwfunc% = 10 THEN GOSUB Setx: GOTO 23520
 IF BB$ = CHR$(13) AND rwfunc% = 12 THEN GOSUB Sety: GOTO 23520
 IF BB$ = CHR$(13) AND rwfunc% = 14 THEN GOSUB Setdir: GOTO 23520
 IF BB$ = CHR$(13) AND rwfunc% = 16 THEN GOSUB Seteq: GOTO 23520
 IF BB$ = CHR$(13) AND rwfunc% = 18 THEN GOSUB Setfilenumber: GOTO 23520




'IF BB$ = CHR$(13) AND rwfunc% = 18 THEN GOSUB ViewResults:  CLS : Retflag% = 1: GOTO
CalcSound

 IF BB$ = "" THEN GOTO 23520
 GOTO 23520

23700 '**** down key pressed
 LOCATE rwfunc%, 18, 0
 PRINT "    ";
 rwfunc% = rwfunc% + 2
 IF rwfunc% > 20 THEN rwfunc% = 20
23760 LOCATE rwfunc%, 18, 0
 PRINT "--> ";
 RETURN
23800 '**** up key pressed
 LOCATE rwfunc%, 18, 0
 PRINT "    ";
 rwfunc% = rwfunc% - 2
 IF rwfunc% < 10 THEN rwfunc% = 10
 LOCATE rwfunc%, 18, 0
 PRINT "--> ";
 RETURN

RETURN


Setx:
  OutFlag% = 0
  ROWIN% = 10: COLIN% = 46
  GOSUB linein
  IF OutFlag% > 0 THEN RETURN
  IF linein$ <> "" THEN xxo# = VAL(linein$)
  GOSUB refresh7
RETURN

Sety:
  OutFlag% = 0
  ROWIN% = 12: COLIN% = 46
  GOSUB linein
  IF OutFlag% > 0 THEN RETURN
  IF linein$ <> "" THEN yyo# = VAL(linein$)
  GOSUB refresh7
RETURN
```

```
Setdir:
  OutFlag% = 0
  ROWIN% = 14: COLIN% = 46
  GOSUB linein
  IF OutFlag% > 0 THEN RETURN
  IF linein$ <> "" THEN po# = VAL(linein$)
  GOSUB refresh7
RETURN

Seteq:
  OutFlag% = 0
  ROWIN% = 16: COLIN% = 46
  GOSUB linein
  IF OutFlag% > 0 THEN RETURN
  IF linein$ <> "" THEN EQN% = VAL(linein$)
  GOSUB refresh7
RETURN


Setfilenumber:
 OutFlag% = 0
  ROWIN% = 18: COLIN% = 46
  GOSUB linein
  IF OutFlag% > 0 THEN RETURN
  IF linein$ <> "" THEN fun% = VAL(linein$)
  GOSUB refresh7
endin:
RETURN



refresh7:

FOR Lc% = 2 TO 4
LOCATE Lc%, 1
PRINT SPACE$(79)
NEXT Lc%


LOCATE 10, 25, 0: PRINT "Initial x value    ";
 LOCATE 10, 46
 PRINT USING "###.####"; xxo#

LOCATE 12, 25, 0: PRINT "Initial y value     ";
 LOCATE 12, 46
 PRINT USING "###.####"; yyo#

LOCATE 14, 25, 0: PRINT "Initial dir Radians ";
 LOCATE 14, 46
 PRINT USING "###.####"; po#

LOCATE 16, 25, 0: PRINT "Selected Index Eq 2-5";
 LOCATE 16, 46
 PRINT USING "###.####"; EQN%;

LOCATE 18, 25, 0: PRINT "Initial file num";
 LOCATE 18, 46
 PRINT USING "###.####"; fun%;


RETURN


linein:
   LOCATE ROWIN%, COLIN%
```

```
   PRINT ; CHR$(176); : PRINT ; CHR$(176); : PRINT ; CHR$(176); : PRINT ; CHR$(176); :
PRINT ; CHR$(176); : PRINT ; CHR$(176);
   linein$ = ""
LINEIN1:
   AX$ = INKEY$
   IF AX$ = "q" THEN AX$ = "": CLOSE : STOP                   'exit
   IF AX$ = "Q" THEN AX$ = "": CLOSE : STOP
   IF AX$ = "" THEN GOTO LINEIN1  ' ASC(AX$)
   IF AX$ = CHR$(13) THEN GOTO outnow
   'dtff
   IF LEN(AX$) = 2 AND RIGHT$(AX$, 1) = CHR$(59) THEN
  OutFlag% = 4: GOTO outnow
   END IF
   IF LEN(AX$) = 2 AND RIGHT$(AX$, 1) = CHR$(60) THEN
  OutFlag% = 5: GOTO outnow
   END IF
   IF AX$ = CHR$(27) THEN OutFlag% = 1: GOTO outnow  'escape source
   IF LEN(AX$) = 2 THEN
  IF RIGHT$(AX$, 1) = "H" THEN     'up arrow

 OutFlag% = 3
  END IF
  GOTO outnow

   END IF
   IF AX$ = CHR$(8) THEN
 IF LEN(linein$) THEN linein$ = LEFT$(linein$, LEN(linein$) - 1)
ELSE
 linein$ = linein$ + AX$
   END IF

LOCATE ROWIN%, COLIN%
   PRINT ; CHR$(176); : PRINT ; CHR$(176); : PRINT ; CHR$(176); : PRINT ; CHR$(176); :
PRINT ; CHR$(176); : PRINT ; CHR$(176);
LOCATE ROWIN%, COLIN%
   PRINT linein$
   GOTO LINEIN1
outnow:
LOCATE ROWIN%, COLIN%
PRINT ; "      "
RETURN
```