Kerr Black Hole

```
Manipulate[


   If[
    ! slidersEnabled,

    {pT, aI, rI, iL, θI, pθI, frame, tailLength, zoomManual} = presetValues;

    slidersEnabled = True;
     ];



 viewRadius = 10;

   viewθ    = 0.85 π / 2;
   viewϕ    = 0.35 π / 2;
   divergence = 0.05 π / 2;

   rightViewPoint =

viewRadius {
Sin[viewθ] Cos[viewϕ],
Sin[viewθ] Sin[viewϕ],
Cos[viewθ]
    };


   leftViewPoint =

viewRadius {
Sin[viewθ] Cos[viewϕ - divergence],
Sin[viewθ] Sin[viewϕ - divergence],
Cos[viewθ]
    };

 Ee =
ε /.
Solve[
    ( - aI^2 pθI^2 + 2 iL^2 rI +
            2 pθI^2 rI - aI^2 rI^2 -
            iL^2 rI^2 - pθI^2 rI^2 +
            2 rI^3 - rI^4 -
            4 aI iL rI ε + 2 aI^2 rI ε^2 +
            aI^2 rI^2 ε^2 + rI^4 ε^2 +
```

```
          aI^2 (aI^2 + (-2 + rI) rI) (-1 + ε^2) Cos[θI]^2 -
          iL^2 (aI^2 + (-2 + rI) rI) Cot[θI]^2) == 0
  ,
  ε
  ][[2]];


  Ce =
pθI^2 + Cos[θI]^2 (aI^2 (1 - Ee^2) + iL^2 / Sin[θI]^2);


  dynamicEquations =
   {
    r'[τ] == pr[τ] (a^2 - 2 r[τ] + r[τ]^2) / (a^2 Cos[θ[τ]]^2 + r[τ]^2),

pr'[τ] == (a^4 (-a Ee + L)^2 Cos[θ[τ]]^2 + a^4 (L^2 Cos[θ[τ]]^2 Cot[θ[τ]]^2 + pθ[τ]^2) r[τ] +
        a^2 (-a^2 Ee^2 + 2 a Ee L - L^2 + 2 a Ee (a Ee + L) Cos[θ[τ]]^2 - 4 L^2 Cot[θ[τ]]^2 - 4 pθ[τ]^2)
         r[τ]^2 + (4 a^2 Ee^2 - 8 a Ee L + 4 L^2 - 4 a^2 Ee^2 Cos[θ[τ]]^2 +
           4 L^2 Cot[θ[τ]]^2 + 2 a^2 L^2 Cot[θ[τ]]^2 + 2 (2 + a^2) pθ[τ]^2) r[τ]^3 +
        (-2 a^2 Ee^2 + 6 a Ee L - 4 L^2 + a^2 Ee^2 Cos[θ[τ]]^2 - 4 L^2 Cot[θ[τ]]^2 - 4 pθ[τ]^2) r[τ]^4 +
        (L^2 Csc[θ[τ]]^2 + pθ[τ]^2) r[τ]^5 - Ee^2 r[τ]^6 +
        pr[τ]^2 (a^2 - 2 r[τ] + r[τ]^2)^2 (a^2 Cos[θ[τ]]^2 - r[τ]^2 + a^2 r[τ] Sin[θ[τ]]^2)) /
      ((a^2 Cos[θ[τ]]^2 + r[τ]^2)^2 (a^2 - 2 r[τ] + r[τ]^2)^2),

ϕ'[τ] == (a^2 L Cot[θ[τ]]^2 + 2 (a Ee - L - L Cot[θ[τ]]^2) r[τ] + L Csc[θ[τ]]^2 r[τ]^2) /
      ((a^2 Cos[θ[τ]]^2 + r[τ]^2) (a^2 - 2 r[τ] + r[τ]^2)),

θ'[τ] == pθ[τ] / (a^2 Cos[θ[τ]]^2 + r[τ]^2),

pθ'[τ] ==
     ((2 a^2 Cos[θ[τ]] ((Ce + a^2 (-1 + Ee^2) Cos[θ[τ]]^2 - L^2 Cot[θ[τ]]^2) (a^2 - 2 r[τ] + r[τ]^2) -
           (Ce + (-a Ee + L)^2 + r[τ]^2) (a^2 - 2 r[τ] + r[τ]^2) + (a L - Ee (a^2 + r[τ]^2))^2)
         Sin[θ[τ]]) / (a^2 - 2 r[τ] + r[τ]^2) - a^2 pθ[τ]^2 Sin[2 θ[τ]] - a^2 pr[τ]^2
        (a^2 - 2 r[τ] + r[τ]^2) Sin[2 θ[τ]] + (a^2 Cos[θ[τ]]^2 + r[τ]^2) (2 L^2 Cot[θ[τ]] +
          2 L^2 Cot[θ[τ]]^3 - a^2 (-1 + Ee^2) Sin[2 θ[τ]])) / (2 (a^2 Cos[θ[τ]]^2 + r[τ]^2)^2)
   };

  initialConditions =
   {
    r[0] == rI,
pr[0] == 0,
θ[0] == θI,
pθ[0] == pθI,
ϕ[0] == 0
```

```
    };

Quiet[
HamiltonianSolve =
NDSolve[
    {

dynamicEquations,

initialConditions

          } /. {a → aI, L → iL},
    {r, ϕ, Θ, pr, pΘ},
    {τ, 0, pT},
         Method → {EventLocator, "Event" → (r[τ] - 1.02 holeSize)}
    ];
  ];

 domain   =   (r /. HamiltonianSolve[[1, 1]])["Domain"];

   {begin, end} =   domain[[1]];

   planetHasPlunged =

    Abs[
      (r[end] /. HamiltonianSolve)[[1]] - holeSize
      ] ≤ 0.05 holeSize;

   startPlot =
    If[
     (end - tailLength) ≤ 0
     ,
     0
     ,
     (end - tailLength)
     ];

   If[
    zoomManual == False,

    initialOuterRadius = (r[end] /. HamiltonianSolve)[[1]];

    frameCantidate =

     1.05

      If[
       initialOuterRadius > rI
       ,
       initialOuterRadius
       ,
       rI
      ];
```

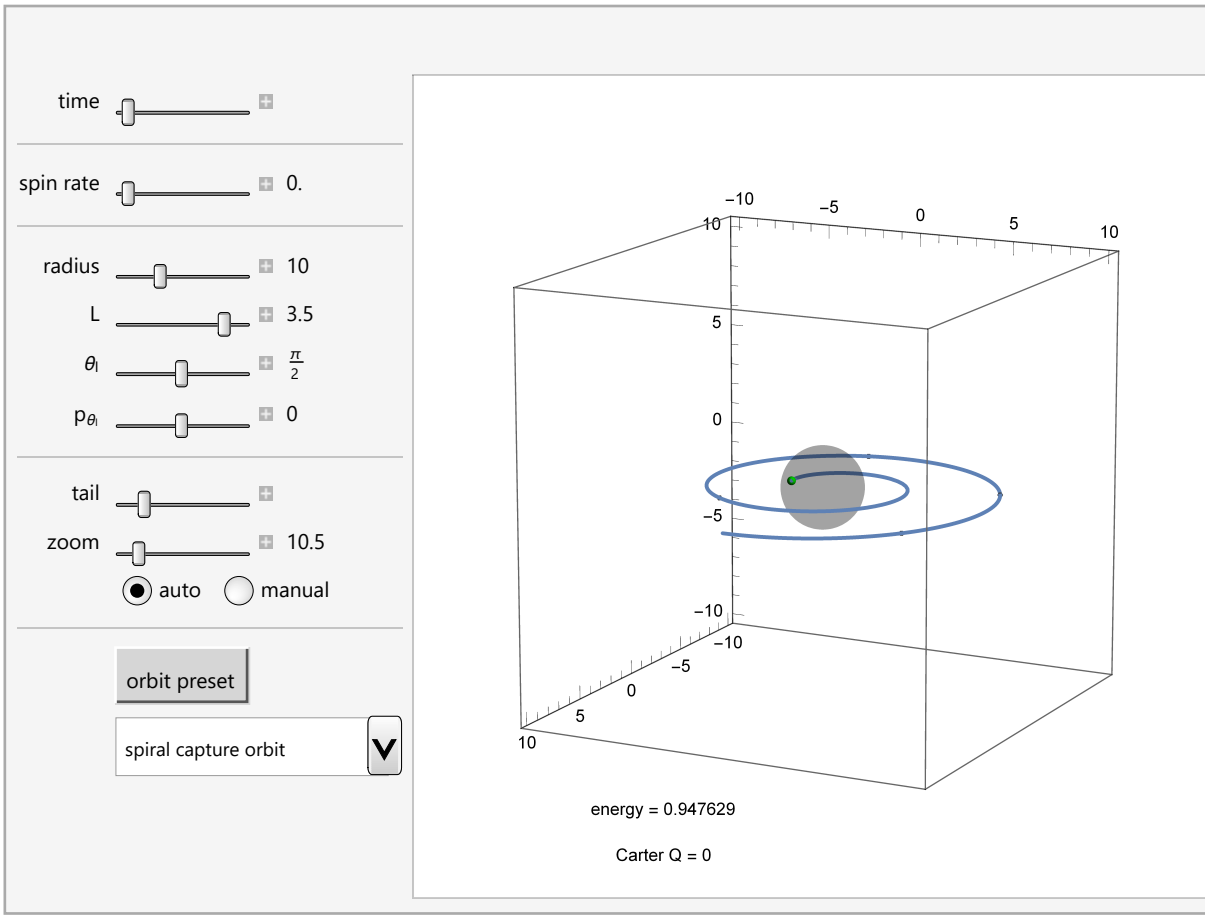
```
     frame =
      If[
       frameCantidate > frame
       ,
       frameCantidate
       ,
       frame
      ];,
     {}];

    planetPosition =
         {
        r[end] Sin[θ[end]] Cos[ϕ[end]],
        r[end] Sin[θ[end]] Sin[ϕ[end]],
        r[end] Cos[θ[end]]
         } /. HamiltonianSolve;

    orbitPlot =

ParametricPlot3D[
       {
            r[τ] Sin[θ[τ]] Cos[ϕ[τ]],
        r[τ] Sin[θ[τ]] Sin[ϕ[τ]],
        r[τ] Cos[θ[τ]]
        } /. HamiltonianSolve,
        {τ, startPlot, end},
        PlotRange →
            {
             {-frame, frame},
             {-frame, frame},
             {-frame, frame}
            },
        PerformanceGoal → "Speed",
        PlotPoints → 200,
        MaxRecursion → 8, (*
                  ViewPoint→rightViewPoint,*)
        SphericalRegion → True,
            Mesh → 4,
            Ticks → Automatic
       ];

    holeSize = 1 + Sqrt[ 1 - aI^2] ;

    planetSize = 0.02 frame;

    If[
planetHasPlunged
  ,
adjustedPlanetSize = 0;
    ,
adjustedPlanetSize = planetSize;
  ];
    )
```

```
 planetGraphic =

   Graphics3D[{Green , Sphere[ planetPosition, adjustedPlanetSize]}];

 noReturnHorizon =

   Graphics3D[{Black  , Sphere[{0, 0, 0}    , holeSize ]}];

 outerErgosphereLimit =

Graphics3D[{
     Black,
     Opacity[0.2],
     Scale[
      Sphere[],
      {2, 2, holeSize},
      {0, 0, 0}
        ]}
       ];


 rightImage =
    Show[

  orbitPlot,
     noReturnHorizon,
     outerErgosphereLimit,
     planetGraphic,

     Graphics3D[Text[StringForm["energy = ``", Ee], {1.5 frame, 0, -1.1 frame}]],

     Graphics3D[
      Text[StringForm["Carter Q = ``", Chop[Ce]], {1.5 frame, 0, -1.3 frame}]],

     ViewPoint → rightViewPoint,
ImageSize → {400, 400}
   ];

 leftImage =
   Show[

  orbitPlot,
    noReturnHorizon,
    outerErgosphereLimit,
    planetGraphic,

    Graphics3D[Text[StringForm["energy = ``", Ee], {1.5 frame, 0, -1.1 frame}]],

    Graphics3D[
     Text[StringForm["Carter Q = ``", Chop[Ce]], {1.5 frame, 0, -1.3 frame}]],

    ViewPoint → leftViewPoint,
ImageSize → {400, 400}
  ]
```

```
 ,

{
 {pT, tailLength, "time"}, 150, 1200,
 ImageSize → Tiny,
AnimationRate → 3,
 DisplayAllSteps → False,
  DefaultDuration → 15,
  ControlPlacement → Left
},

Delimiter,

{
{aI, 0.99, "spin rate"}, 0, 0.99, .01,
Appearance → "Labeled",
ImageSize → Tiny,
  ControlPlacement → Left
},

Delimiter,

{
{rI, 4, "radius"}, 2.1, 30, .01,
Appearance → "Labeled",
ImageSize → Tiny,
  ControlPlacement → Left
},

{
{iL, 2, "L"}, -4.5, 4.5, .01,
Appearance → "Labeled",
ImageSize → Tiny,
  ControlPlacement → Left
},

{
  {θI, π/3, Subscript["θ", "I"]}, π / 7, 6 π / 7, π/210,
Appearance → "Labeled",
ImageSize → Tiny,
  ControlPlacement → Left
},

{
{pθI, 0.76, Subscript["p", Subscript["θ", "I"]]}, -3, 3, .01,
Appearance → "Labeled",
ImageSize → Tiny,
  ControlPlacement → Left
},

Delimiter,


 {
  {tailLength, 1200, "tail"}, 150, 1500,
```

```
  ControlPlacement → Left,
  ImageSize → Tiny
 },

{
{frame, 4.5, "zoom"}, 2.5, 100, .01,
Appearance → "Labeled",
ImageSize → Tiny,
  Enabled → zoomManual,
  ControlPlacement → Left
},

 {
  {zoomManual, False, ""},
  {False → "auto", True → "manual"},
  ControlType → RadioButton,
  ControlPlacement → Left
 },




Delimiter,

 {
  {slidersEnabled, True, ""},
  {False → "orbit preset"},
  ControlType → Setter,
  ImageSize → Tiny,
  ControlPlacement → Left
 },



 {{presetValues, {1200,  0.99, 4, 2, π/3, 0.767851, 4.5, 1200, False}, ""},
  {
   {300,  0.9, 4, 2.148, 1.037, 0, 4.2, 350, False} →
    Style["closed orbit                        ", 10] ,
   {1200,  0.99, 4, 2, π/3, 0.767851, 4.5, 1200, False} →
    Style["constant radius orbit              ", 10] ,
   {150,  0.0, 10, 3.5, π/2, 0, 4.5, 350, False} →
    Style["spiral capture orbit               ", 10] ,
   {150,  0.0,  4,   3.99999, π/2, 0, 4.5, 350, False} →
    Style["unstable circular orbit capture   ", 10] ,
   {100,  0.0,  4,   4.00001, π/2, 0, 4.5, 350, False} →
    Style[ "unstable circular orbit escape   ", 10] ,
   {330, 0.99,  25, 2.427, π/2, 0, 25, 330, False} →
    Style[ "equatorial (1,1,1) zoom and whirl orbit"] ,
   {150,  0.9,   4,   - 4.5, π/2, 0, 4.2, 350, False} →
    Style["orbit reverse and capture        ", 10],
   {150,  0.99,   10,  1.05769, π/2, 2.89, 4, 150, True} →
    Style["3D zoom and whirl orbit          ", 10]
  },
  ControlType → PopupMenu,
```

```
  ControlPlacement → Left,
  ImageSize → Small
 },

Style[
"",
Bold, Small
 ],


SynchronousUpdating → False,

SaveDefinitions → True,

TrackedSymbols → Manipulate,

AutorunSequencing → {1, 2, 3, 4, 6, 7}
]
```



time

spin rate 0.99

radius 4

L 2

θ_I $\frac{\pi}{3}$

p_{θ_I} 0.76

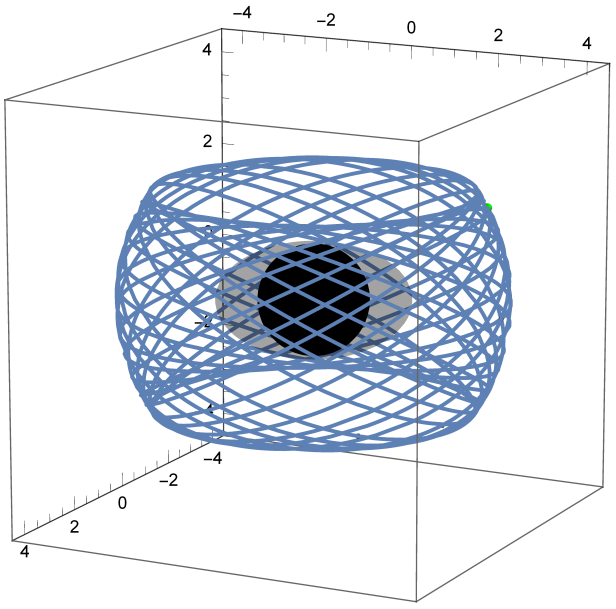
tail

zoom 4.5

auto manual

orbit preset

3D zoom and whirl orbit



energy = 0.890181

Carter Q = 1.9618