```
(*  Lens Distortion Mathematica Notebook. Shows step-
  by-step how Mathematica 9 was used
    to derive the formulas covered in the http://
 www.decarpentier.nl/lens-distortion.
    See that page for more details and sample applications.
    Giliam de Carpentier. Copyright 2015. *)
```

In[28]:=
```
ClearAll
```

Out[28]=
```
ClearAll
```

In[29]:=
```
(* Define the mapping from unnormalized perspective
 to unnormalized stereographic: 2tan(atan(k)/2) *)
up2us = 2 {pux, puy} / (1 + Sqrt[1 + pux^2 + puy^2 ])
us2up = 4 {sux, suy} / (4 - (sux^2 + suy^2 ))
```

Out[29]=
$$\left\{\frac{2\,\text{pux}}{1+\sqrt{1+\text{pux}^2+\text{puy}^2}},\ \frac{2\,\text{puy}}{1+\sqrt{1+\text{pux}^2+\text{puy}^2}}\right\}$$

Out[30]=
$$\left\{\frac{4\,\text{sux}}{4-\text{sux}^2-\text{suy}^2},\ \frac{4\,\text{suy}}{4-\text{sux}^2-\text{suy}^2}\right\}$$

In[31]:=
```
(* prove that up2us and us2up are eachother's inverse *)
maxsusquared =
  Limit[Limit[First[up2us]^2 + Last[up2us]^2, pux → Infinity], puy → Infinity];
FullSimplify[up2us /. pux → First[us2up] /. puy → Last[us2up],
 0 < sux^2 + suy^2 < maxsusquared]
```

Out[32]=
```
{sux, suy}
```

In[33]:=
```
(* Demonstrate that re-mapping a perspective projected position on the
  unit sphere with up2us is indeed equivalent to mapping that position
  directly with the stereographic projection formula 2*{x, y}/(1 + z). *)
FullSimplify[up2us /. pux → x / z  /. puy → y / z,  z > 0] /. Sqrt[x^2 + y^2 + z^2] → 1
```

Out[33]=
$$\left\{\frac{2\,x}{1+z},\ \frac{2\,y}{1+z}\right\}$$

In[34]:=
```
(* Define the mapping from normalized
  perspective to unnormalized stereographic. h = tan(FOVy/2),
a=aspectRatio, c=cylindricalRatio, diag=tan(FOVdiag/2) *)
p2us = FullSimplify[up2us /. pux → scale * (a * c) * px /. puy → scale * py /.
    scale -> s * diag / Sqrt[1 + (a * c)^2] /. diag → h * Sqrt[1 + a^2]]
```

Out[34]=
$$\left\{\frac{2\,a\,\sqrt{1+a^2}\ c\,h\,px\,s}{\sqrt{1+a^2\,c^2}\left(1+\sqrt{1+\frac{\left(1+a^2\right)\,h^2\,\left(a^2\,c^2\,px^2+py^2\right)\,s^2}{1+a^2\,c^2}}\right)},\ \frac{2\sqrt{1+a^2}\ h\,py\,s}{\sqrt{1+a^2\,c^2}\left(1+\sqrt{1+\frac{\left(1+a^2\right)\,h^2\,\left(a^2\,c^2\,px^2+py^2\right)\,s^2}{1+a^2\,c^2}}\right)}\right\}$$

In[35]:=
```
(* Scale the output of p2us to get a direct solution for
 normalized perspective to normalized stereographic coordinates *)
p2sfull = FullSimplify[p2us / (p2us /. px → 1 /. py → 1), {a > 0, c > 0, h > 0, s > 0}]
```

Out[35]=
$$\left\{\frac{px\left(1+\sqrt{1+\left(1+a^2\right)h^2s^2}\right)}{1+\sqrt{1+\frac{\left(1+a^2\right)h^2\left(a^2c^2px^2+py^2\right)s^2}{1+a^2c^2}}},\ \frac{py\left(1+\sqrt{1+\left(1+a^2\right)h^2s^2}\right)}{1+\sqrt{1+\frac{\left(1+a^2\right)h^2\left(a^2c^2px^2+py^2\right)s^2}{1+a^2c^2}}}\right\}$$

In[36]:=
```
(* Define the n and z helper constants used below *)
paramsZ = (1 + Sqrt[1 + (1 + a^2) h^2 s^2]) / 2;
paramsN = (z - 1) / (1 + (a c)^2);
params = {z → paramsZ, n → (paramsN /. z -> paramsZ)};
```

In[39]:=
```
(* Redefine the normalized-perspective-to-
 normalized-stereographic formula in terms of n and z
  and show that this formula is equivalent to p2sSolved *)
p2s = 2 z {px, py} / (1 + Sqrt[1 + 4 n z (a^2 c^2 px^2 + py^2)])
FullSimplify[p2sfull == p2s /. params]
```

Out[39]=
$$\left\{\frac{2\,px\,z}{1+\sqrt{1+4\,n\left(a^2c^2px^2+py^2\right)z}},\ \frac{2\,py\,z}{1+\sqrt{1+4\,n\left(a^2c^2px^2+py^2\right)z}}\right\}$$

Out[40]= True

In[41]:=
```
(* Inverse p2s to get s2p, which thus converts normalized
 stereographic coordinates to normalized perspective coordinates *)
s2p = {px, py} /. First[FullSimplify[Solve[p2s == {sx, sy}, {px, py}]]]
```

Out[41]=
$$\left\{\frac{sx}{-n\left(a^2c^2sx^2+sy^2\right)+z},\ \frac{sy}{-n\left(a^2c^2sx^2+sy^2\right)+z}\right\}$$

In[42]:=
```
(* define the direct unit sphere to perspective
 and unit sphere to stereographic projections *)
Perspective[w_] := {w[[1]] / w[[3]], w[[2]] / w[[3]]}
Project[w_] :=
 Perspective[w] * 2 / (1 + Sqrt[1 + s^2 * Perspective[w].Perspective[w]])
```

In[44]:=
```
(* prepare the points q0:(0,0), qx:(t,0) and qy:(0, t) on the unit sphere,
which are rotated over the y axis by an angle of θ.
   these points will conceptually serve as one of the
 corners of the screen θ radians away from the screen's center,
which thus makes θ equal to half of diagonal FOV angle*)
r = RotationMatrix[θ, {0, 1, 0}];
q0 = Project[r.{0, 0, 1}];
qx = Project[r.{t, 0, 1}];
qy = Project[r.{0, t, 1}];
lenx = Sqrt[(qx - q0).(qx - q0)];
leny = Sqrt[(qy - q0).(qy - q0)];
```

In[50]:=
```
(* define maxstretch as the limit of the ratio of the
 length of qx and qy as a function of θ as t goes to zero *)
maxstretchfromrad = FullSimplify[Limit[lenx / leny, t → 0], {0 <= θ < Pi / 2, s > 0}]
maxstretchfromtan = FullSimplify[
  maxstretchfromrad /. θ → ArcTan[h * Sqrt[1 + a^2]], {a > 0, h > 0, s > 0}]
FullSimplify[maxstretchfromtan /. (1 + a^2) → ((2 z - 1)^2 - 1) / (h^2 s^2),
 {s > 0, z > 1}]

maxstretchfromz = Sqrt[1 + 4 (z - 1) z / s^2] / (2 z - 1)

FullSimplify[
 maxstretchfromtan == maxstretchfromz /. z -> paramsZ, {a > 0, h > 0, s > 0}]
```

Out[50]=
$$\frac{\text{Sec}[\theta]}{\sqrt{1 + s^2\,\text{Tan}[\theta]^2}}$$

Out[51]=
$$\sqrt{\frac{1 + \left(1 + a^2\right) h^2}{1 + \left(1 + a^2\right) h^2 s^2}}$$

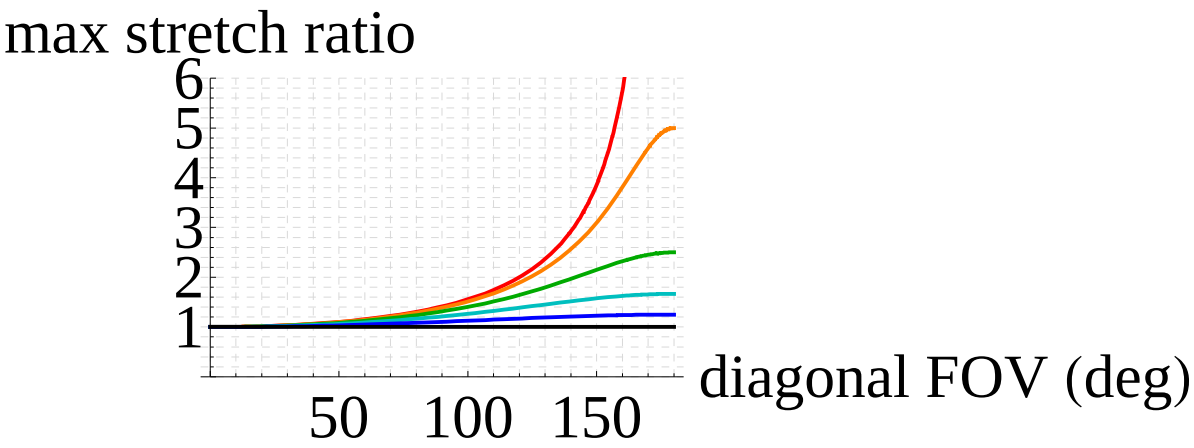
Out[52]=
$$\sqrt{\frac{s^2 + 4\,(-1 + z)\,z}{(s - 2\,s\,z)^2}}$$

Out[53]=
$$\frac{\sqrt{1 + \frac{4\,(-1+z)\,z}{s^2}}}{-1 + 2\,z}$$

Out[54]= True

In[129]:=
```
maxstretchfromrad;
{% /. s → 0.0, % /. s → 0.2, % /. s → 0.4,
   % /. s → 0.6, % /. s → 0.8, % /. s → 1} /. θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {0, 6}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 6, .2}]},
  GridLinesStyle → Directive[LightGray, Dashed],
  AxesLabel → {"diagonal FOV (deg)", "max stretch ratio"}, ImageSize → Full,
  PlotStyle → { {Thick, Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Out[131]=



In[58]:=
```
(* define maxscale as the limit of the
 square root of the area formed by the q0, qx,
qy and qx+qy-q0 relative to t itself as t goes to 0 *)
maxscalefromrad =
 FullSimplify[Limit[Sqrt[lenx * leny / t^2], t → 0], {0 <= θ < Pi / 2, s > 0}]
maxscalefromtan = FullSimplify[
  maxscalefromrad /. θ → ArcTan[h * Sqrt[1 + a^2]], {a > 0, h > 0, s > 0}]
FullSimplify[maxscalefromtan /. (1 + a^2) → ((2 z - 1) ^2 - 1) / (h^2 s^2),
 {s > 0, z > 1}]
maxscalefromz = (s^2 + 4 (-1 + z) z)^(3/4) / (s (2 s z - s)^(1/2) z)
FullSimplify[
 maxscalefromtan^4 == maxscalefromz^4 /. z → paramsZ, {a > 0, h > 0, s > 0}]
```

Out[58]=
$$\frac{2\,\text{Sec}[\theta]^{3/2}}{\left(1+s^2\,\text{Tan}[\theta]^2\right)^{1/4}+\left(1+s^2\,\text{Tan}[\theta]^2\right)^{3/4}}$$

Out[59]=
$$\frac{2\left(1+\left(1+a^2\right)h^2\right)^{3/4}}{\left(1+\left(1+a^2\right)h^2 s^2\right)^{1/4}+\left(1+\left(1+a^2\right)h^2 s^2\right)^{3/4}}$$

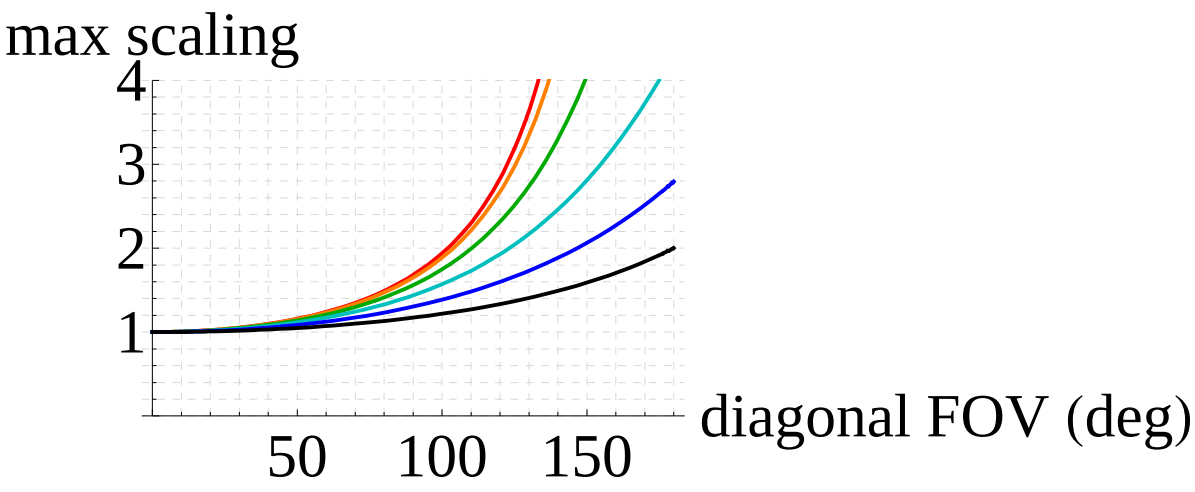
Out[60]=
$$\frac{\left(\frac{s^2+4\,(-1+z)\,z}{s^2}\right)^{3/4}}{z\sqrt{-1+2z}}$$

Out[61]=
$$\frac{\left(s^2+4\,(-1+z)\,z\right)^{3/4}}{s\,z\sqrt{-s+2sz}}$$

Out[62]= True

In[126]:=
```
maxscalefromrad;
{% /. s → 0.0, % /. s → 0.2, % /. s → 0.4,
   % /. s → 0.6, % /. s → 0.8, % /. s → 1} /. θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {0, 4}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 5, .2}]},
  GridLinesStyle → Directive[LightGray, Dashed],
  AxesLabel → {"diagonal FOV (deg)", "max scaling"}, ImageSize → Full,
  PlotStyle → { {Thick, Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Out[128]=



In[66]:=
```
(* define maxcurvature as the curvature of the limit of the
 curve from q0 to qy as t goes to 0, where curvature is defined
 by the standard formula (dx*ddy * dy*ddx) / (ddx + ddy)^3/2  *)
(* relmaxcurvature represents the screendiagonal relative to
 the diameter of a circle with maxcurvature. as a result, *)
(* 1.0 means that an on-screen line can be bent so much
  that's curvature is equivalent to the curvature of a
  circle going through all the corners of the screen *)
maxcurvaturefromrad = FullSimplify[
   (v[[1]] * a[[2]] - v[[2]] * a[[1]]) / ((v[[1]]^2 + v[[2]]^2)^(3 / 2)) /.
     {v → FullSimplify[D[(qy - q0), t]], a → FullSimplify[D[(qy - q0), {t, 2}]]} /.
    t → 0, {0 <= θ < Pi / 2, s > 0}];

relmaxcurvaturefromrad = FullSimplify[
  screendiagonal / mincurvaturediameter /. {screendiagonal → 2 Norm[q0],
    mincurvaturediameter → 2 / maxcurvaturefromrad}, {0 <= θ < Pi / 2, s > 0}]
relmaxcurvaturefromtan = FullSimplify[relmaxcurvaturefromrad /.
   θ → ArcTan[h * Sqrt[1 + a^2]], {a > 0, h > 0, s > 0}]

relmaxcurvaturefromz = Factor[FullSimplify[relmaxcurvaturefromtan /.
    (1 + a^2) → ((2 z - 1)^2 - 1) / (h^2 s^2), {s > 0, z > 1}] ]
```

Part::partd : Part specification v⟦1⟧ is longer than depth of object. ≫

Part::partd : Part specification a⟦2⟧ is longer than depth of object. ≫

Part::partd : Part specification v⟦2⟧ is longer than depth of object. ≫

General::stop : Further output of Part::partd will be suppressed during this calculation. ≫

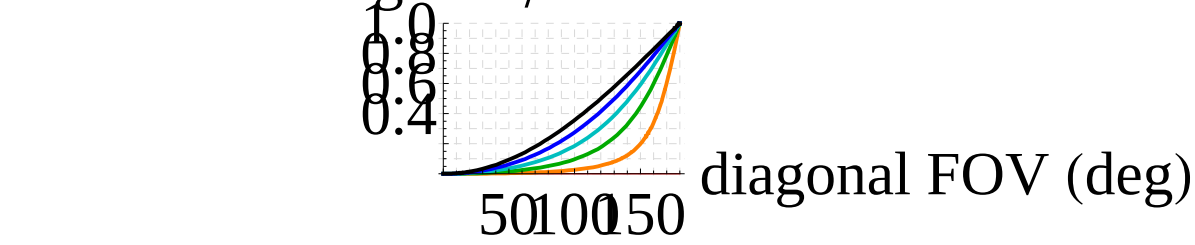
Out[67]=
$$\frac{s^2 \operatorname{Tan}[\theta]^2}{\sqrt{\left(1 + s^2 \operatorname{Tan}[\theta]^2\right)\left(2 + s^2 \operatorname{Tan}[\theta]^2 + 2\sqrt{1 + s^2 \operatorname{Tan}[\theta]^2}\right)}}$$

Out[68]=
$$\left(\left(1 + a^2\right) h^2 s^2\right) \Big/ \left(\sqrt{\left(1 + \left(1 + a^2\right) h^2 s^2\right)\left(2 + \left(1 + a^2\right) h^2 s^2 + 2\sqrt{1 + \left(1 + a^2\right) h^2 s^2}\right)}\right)$$

Out[69]=
$$\frac{2\,(-1 + z)}{-1 + 2z}$$

In[123]:=
```
relmaxcurvaturefromrad;
{% /. s → 0.0, % /. s → 0.2, % /. s → 0.4,
   % /. s → 0.6, % /. s → 0.8, % /. s → 1} /. θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {-.01, 1}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 2, .1}]},
  GridLinesStyle → Directive[LightGray, Dashed],
  AxesLabel → {"diagonal FOV (deg)", "max screen diagonal / line diameter"},
  ImageSize → Full,
  PlotStyle → { {Thickness[0.005], Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Out[125]=



In[73]:=
```
(* coveredarea is the relative area in perspective space of the
  screen rectangle in distorted space. this can be used to calculate
  how much of the original perspective-projected screen will be *)
(* sampled when applying s2p on a full screen, so,
1.0 means that all of the original perspective-projected image is used,
while 0.0 means that the whole distorted screen will be filled only by *)
(* the color of the center pixel of the original perspective-
 projected image *)
rightmostpxfrompy = FullSimplify[
  First[s2p] /. Last[Solve[Last[s2p] == py, sy]] /. sx → 1]
topmostpyfrompx = FullSimplify[
  Last[s2p] /. Last[Solve[First[s2p] == px, sx]] /. sy → 1]
FullSimplify[(rightmostpxfrompy /. py → t /. n → paramsN) ==
  (topmostpyfrompx /. px → t /. n → paramsN /. a → 1 / a /. c → 1 / c)]

FullSimplify[n (a^2 c^2 n - z) < 0 /. params, {a > 0, c > 0, h > 0, 0 < s < 1}]
unusedarealeftright = FullSimplify[
  1 - (Integrate[rightmostpxfrompy, {py, 0, 1}]), Sqrt[n (a^2 c^2 n - z)] ∉ Reals]
substitudec = {c^2 -> ((z - 1) / n - 1) / a^2 };
FullSimplify[c^2 == (c^2 /. substitudec) /. params]

FullSimplify[unusedarealeftright /. substitudec, {z > 1, n > 0, a > 0, c > 0}]
FullSimplify[
 % /. ArcSinh[2 Sqrt[n (1 + n)]] → 2 Sqrt[n (1 + n)] RelArcSinh[2 Sqrt[n (1 + n)]], n > 0]

coveredarea = FullSimplify[1 / 2 - %] + FullSimplify[1 / 2 - % /. n → m]
coveredareacalc = coveredarea /.
      RelArcSinh[2 Sqrt[m (1 + m)]] -> ArcSinh[2 Sqrt[m (1 + m)]] / (2 Sqrt[m (1 + m)]) /.
     RelArcSinh[2 Sqrt[n (1 + n)]] -> ArcSinh[2 Sqrt[n (1 + n)]] / (2 Sqrt[n (1 + n)]) /.
   m → (paramsN /. a → 1 / a /. c → 1 / c);

example = {a -> 16 / 9, c → 1.2, h → 1.4, s → 0.6};
FullSimplify[(coveredareacalc /. params /. example) ==
  (1 - Integrate[1 - rightmostpxfrompy /. params /. example, {py, 0, 1}] -
    Integrate[1 - topmostpyfrompx /. params /. example, {px, 0, 1}])]
```

Out[73]=
$$\frac{2\, n\, py^2}{-1+\sqrt{1+4\, n\, py^2\, \left(-a^2\, c^2\, n+z\right)}}$$

Out[74]=
$$\frac{2\, a^2\, c^2\, n\, px^2}{-1+\sqrt{1+4\, a^2\, c^2\, n\, px^2\, (-n+z)}}$$

Out[75]= True

Out[76]= True

Out[77]=
$$\text{ConditionalExpression}\left[1+\left(2\sqrt{n}\ \sqrt{a^2c^2n-z}\ \left(2+\sqrt{1+4n\left(-a^2c^2n+z\right)}\right)+\text{ArcSin}\left[2\sqrt{n}\ \sqrt{a^2c^2n-z}\ \right]\right)\Big/\left(8\sqrt{n}\ \left(a^2c^2n-z\right)^{3/2}\right),\ \text{Im}\left[\frac{1}{\sqrt{4a^2c^2n^2-4nz}}\right]\neq 0\ ||\ \text{Re}\left[\frac{1}{\sqrt{4a^2c^2n^2-4nz}}\right]>1\ ||\ \left(\left(\text{Re}\left[\frac{1}{\sqrt{4a^2c^2n^2-4nz}}\right]\geq 1\ ||\ \text{Re}\left[\frac{1}{\sqrt{4a^2c^2n^2-4nz}}\right]\leq 0\right)\&\&\left(\text{Im}\left[n\left(a^2c^2n-z\right)\right]\neq 0\ ||\ \text{Re}\left[n\left(a^2c^2n-z\right)\right]<0\right)\right)\right]$$

Out[79]= True

Out[80]=
$$\frac{2n(1+n)(1+2n)-\sqrt{n(1+n)}\ \text{ArcSinh}\left[2\sqrt{n(1+n)}\ \right]}{8n(1+n)^2}$$

Out[81]=
$$\frac{1+2n-\text{RelArcSinh}\left[2\sqrt{n(1+n)}\ \right]}{4+4n}$$

Out[82]=
$$\frac{1+\text{RelArcSinh}\left[2\sqrt{m(1+m)}\ \right]}{4+4m}+\frac{1+\text{RelArcSinh}\left[2\sqrt{n(1+n)}\ \right]}{4+4n}$$

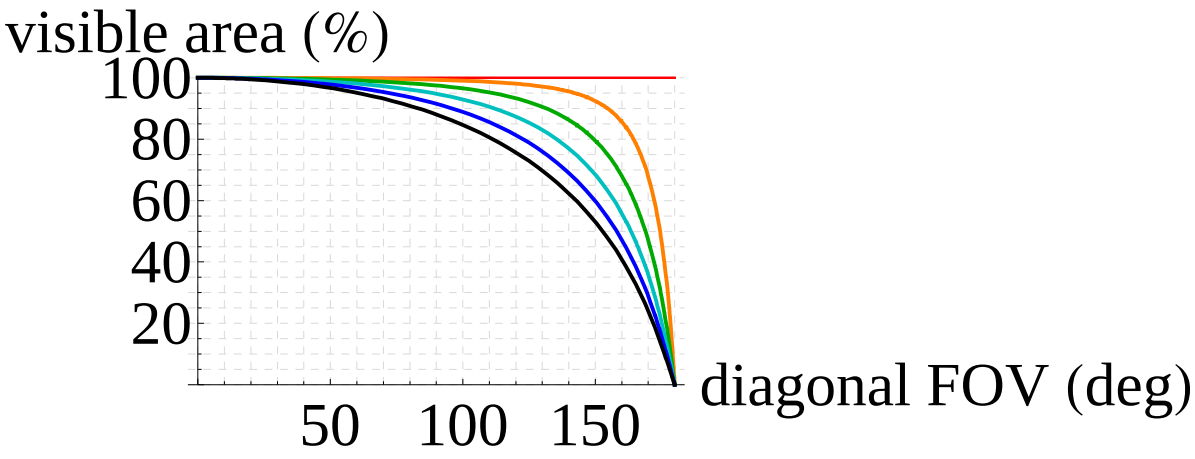
Out[85]= True

In[86]:=
```
topmostpyfrompx =
  FullSimplify[Last[s2p] /. Last[Solve[First[s2p] == px, sx]] /. sy → 1];
test1 = Integrate[
   1 - rightmostpxfrompy /. params /. c → 1 /. s → 0.6 /. h → Tan[θ] / Sqrt[1 + a^2] /.
      θ → deg / 2 * Degree /. deg → 150 /. a → 16 / 9, {py, 0, 1}];
test2 = Integrate[1 - topmostpyfrompx /. params /. c → 1 /. s → 0.6 /.
       h → Tan[θ] / Sqrt[1 + a^2] /.
      θ → deg / 2 * Degree /. deg → 150 /. a → 16 / 9, {px, 0, 1}];
FullSimplify[1 - test1 - test2 == coveredareacalc /. params /. a → 16 / 9 /. c → 1 /.
      s → 0.6 /. h → Tan[θ] / Sqrt[1 + a^2] /.
    θ → deg / 2 * Degree /. deg → 150 /. a → 16 / 9]
```

Out[89]= True

In[120]:=
```
(* calculate and plot the area visible
  after distortion relative to the original perspective-
 projected screen area when sampling the perspective image using s2p *)
100 * coveredareacalc /. params /. c → 1 /. h → Tan[θ] / Sqrt[1 + a^2] /. a → 16 / 9;
{Limit[%, s → 0.0], % /. s → 0.2, % /. s → 0.4,
   % /. s → 0.6, % /. s → 0.8, % /. s → 1} /. θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {0, 100}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 100, 5}]},
  GridLinesStyle → Directive[LightGray, Dashed],
  AxesLabel → {"diagonal FOV (deg)", "visible area (%)"}, ImageSize → Full,
  PlotStyle → { {Thickness[0.005], Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Out[122]=



In[93]:=

In[116]:=
```
(* calculate and plot the average relative
 remaining resolution (i.e. the average pixel density)
 when sampling the perspective image using s2p *)
relavgresolution = Sqrt[coveredareacalc];
100 * relavgresolution /. params /. c → 2 /. h → Tan[θ] / Sqrt[1 + a^2] /. a → 16 / 9;
{Limit[%, s → 0.0], % /. s → 0.2, % /. s → 0.4,
   % /. s → 0.6, % /. s → 0.8, % /. s → 1} /. θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {0, 100}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 100, 5}]},
  GridLinesStyle → Directive[LightGray, Dashed], AxesLabel →
   {"diagonal FOV (deg)", "relative resolution (%)"}, ImageSize → Full,
  PlotStyle → { {Thickness[0.005], Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Power::infy : Infinite expression $\frac{1}{\sqrt{0.}}$ encountered. ≫

Infinity::indet : Indeterminate expression $0.\sqrt{1105}$ ComplexInfinity encountered. ≫

Power::infy : Infinite expression $\frac{1}{\sqrt{0.}}$ encountered. ≫

Infinity::indet : Indeterminate expression $0.\sqrt{1105}$ ComplexInfinity encountered. ≫

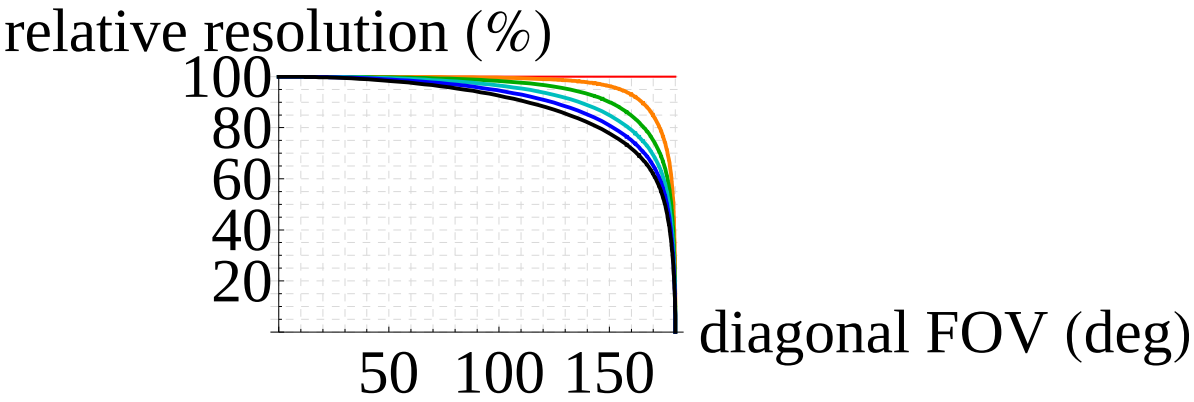
Power::infy : Infinite expression $\frac{1}{\sqrt{0.}}$ encountered. ≫

General::stop : Further output of Power::infy will be suppressed during this calculation. ≫

Infinity::indet : Indeterminate expression $0.\sqrt{1105}$ ComplexInfinity encountered. ≫

General::stop : Further output of Infinity::indet will be suppressed during this calculation. ≫

Out[119]=



In[98]:=

In[99]:=

In[111]:=
```
(* calculate and plot the minimum relative
 remaining resolution (which happens at the image's center)
 when sampling the perspective image using s2p *)
Limit[Limit[s2p / {sx, sy} , sx → 0], sy → 0]
relcenterresolution = First[%];

100 * relcenterresolution /. params /. c → 2 /.
   h → Tan[θ] / Sqrt[1 + a^2] /. a → 16 / 9;
{% /. s → 0.0, % /. s → 0.2, % /. s → 0.4, % /. s → 0.6, % /. s → 0.8, % /. s → 1} /.
  θ → deg / 2 * Degree;
Plot[%, {deg, 0, 180}, {PlotRange → {0, 100}, AspectRatio → 1 / GoldenRatio,
  GridLines → {Table[x, {x, 0, 180, 10}], Table[x, {x, 0, 100, 5}]},
  GridLinesStyle → Directive[LightGray, Dashed], AxesLabel →
   {"diagonal FOV (deg)", "relative resolution (%)"}, ImageSize → Full,
  PlotStyle → { {Thick, Red} , {Thick, Orange}, {Thick, Darker[Green]} ,
    {Thick, Darker[Cyan, 0.25]}, {Thick, Blue}, {Thick, Black}}}]
```

Out[111]= $\left\{\frac{1}{z}, \frac{1}{z}\right\}$

Out[115]=