```
> eq1 := delay
      = 1/light_speed*(sqrt(x^2+(y+h_r/tan(elevation))^2+(h_r)^2)-h_r/sin(elevation)
      -y*cos(elevation))
```

$$eq1 := delay = \frac{1}{light\_speed}\left(\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+(h\_r)^2}-\frac{h\_r}{\sin(elevation)}-y\cdot\cos(elevation)\right)$$

$$eq1 := delay = \frac{\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+h\_r^2}-\frac{h\_r}{\sin(elevation)}-y\cos(elevation)}{light\_speed} \quad (1)$$

```
> eq2 := f_doppler = -v_ty*cos(elevation)-v_tz*sin(elevation)
      +(v_rx*x+v_ry*(y+h_r/tan(elevation))-v_rz*h_r)
      /sqrt(x^2+(y+h_r/tan(elevation))^2+(h_r)^2)
```

$$eq2 := f\_doppler = -v\_ty\cdot\cos(elevation)-v\_tz\cdot\sin(elevation)+\frac{v\_rx\cdot x+v\_ry\cdot\left(y+\frac{h\_r}{\tan(elevation)}\right)-v\_rz\cdot h\_r}{\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+(h\_r)^2}}$$

$$eq2 := f\_doppler = -v\_ty\cos(elevation)-v\_tz\sin(elevation)+\frac{v\_rx\,x+v\_ry\left(y+\frac{h\_r}{\tan(elevation)}\right)-v\_rz\,h\_r}{\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+h\_r^2}} \quad (2)$$

```
> sys := {eq1, eq2}
```

$$sys := \left\{ delay = \frac{1}{light\_speed}\left(\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+h\_r^2}-\frac{h\_r}{\sin(elevation)}-y\cos(elevation)\right),\ f\_doppler = -v\_ty\cos(elevation)-v\_tz\sin(elevation)+\frac{v\_rx\,x+v\_ry\left(y+\frac{h\_r}{\tan(elevation)}\right)-v\_rz\,h\_r}{\sqrt{x^2+\left(y+\frac{h\_r}{\tan(elevation)}\right)^2+h\_r^2}} \right\} \quad (3)$$

```
> sol := solve(sys, {x, y}):
> sol[1]:
```

```
> all_x := allvalues(sol[1]):
> with(CodeGeneration):
> Python(sol_x_1)
cg14 = x == ((math.cos(elevation) ** 2 * v_ty + (v_tz * math.sin
(elevation) + f_doppler) * math.cos(elevation) - v_ry) * math.
sqrt(-(-4 * math.cos(elevation) ** 3 * delay * h_r * v_ty * v_tz
* light_speed + (2 * h_r * light_speed * delay * (v_ty - v_tz) *
(v_ty + v_tz) * math.sin(elevation) + (v_tz + v_ry - v_ty +
v_rz) * (-v_tz + v_ry - v_ty - v_rz) * h_r ** 2 - 4 * delay *
h_r * v_tz * f_doppler * light_speed + light_speed ** 2 * delay
** 2 * (v_ty - v_tz) * (v_ty + v_tz)) * math.cos(elevation) ** 2
+ ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h_r ** 2 + 4 * delay *
h_r * v_ty * f_doppler * light_speed + 2 * delay ** 2 * v_ty *
v_tz * light_speed ** 2) * math.sin(elevation) - 2 * f_doppler *
h_r ** 2 * (v_ry - v_ty) - 2 * light_speed * (-2 * v_tz * v_ty +
v_rz * (v_ry - v_ty)) * delay * h_r + 2 * f_doppler *
light_speed ** 2 * v_ty * delay ** 2) * math.cos(elevation) + (2
* f_doppler * (v_rz + v_tz) * h_r ** 2 - 2 * light_speed * delay
* (v_rx ** 2 + v_ry ** 2 - v_tz * v_rz - v_tz ** 2 - f_doppler *
* 2) * h_r + 2 * delay ** 2 * v_tz * f_doppler * light_speed **
2) * math.sin(elevation) + (v_rz ** 2 + 2 * v_tz * v_rz + v_tz *
* 2 + f_doppler ** 2) * h_r ** 2 + 2 * light_speed * delay *
f_doppler * (v_rz + 2 * v_tz) * h_r - light_speed ** 2 * delay *
* 2 * (v_rx ** 2 + v_ry ** 2 - v_tz ** 2 - f_doppler ** 2)) *
math.sin(elevation) ** 2 * v_rx ** 2) + (h_r * (v_ry - v_ty) *
math.cos(elevation) ** 3 + (-h_r * (v_rz + v_tz) * math.sin
(elevation) - delay * v_tz * light_speed - h_r * f_doppler) *
math.cos(elevation) ** 2 - (v_ry - v_ty) * (delay * light_speed
* math.sin(elevation) + h_r) * math.cos(elevation) +
(light_speed * f_doppler * delay + v_rz * h_r + h_r * v_tz) *
math.sin(elevation) + delay * v_tz * light_speed + h_r *
f_doppler) * v_rx ** 2) / ((v_ty ** 2 - v_tz ** 2) * math.cos
(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin(elevation) +
f_doppler) * math.cos(elevation) ** 3 + (2 * v_tz * math.sin
(elevation) * f_doppler + f_doppler ** 2 - v_rx ** 2 - 2 * v_ry
* v_ty + v_tz ** 2) * math.cos(elevation) ** 2 - 2 * v_ry *
(v_tz * math.sin(elevation) + f_doppler) * math.cos(elevation) +
v_rx ** 2 + v_ry ** 2) / math.sin(elevation) / v_rx
> sol_x_2 := simplify(all_x[2]):
> Python(sol_x_2)
cg15 = x == ((-math.cos(elevation) ** 2 * v_ty + (-v_tz * math.
sin(elevation) - f_doppler) * math.cos(elevation) + v_ry) *
math.sqrt(-(-4 * math.cos(elevation) ** 3 * delay * h_r * v_ty *
v_tz * light_speed + (2 * h_r * light_speed * delay * (v_ty -
v_tz) * (v_ty + v_tz) * math.sin(elevation) + (v_tz + v_ry -
v_ty + v_rz) * (-v_tz + v_ry - v_ty - v_rz) * h_r ** 2 - 4 *
delay * h_r * v_tz * f_doppler * light_speed + light_speed ** 2
* delay ** 2 * (v_ty - v_tz) * (v_ty + v_tz)) * math.cos
(elevation) ** 2 + ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h_r **
2 + 4 * delay * h_r * v_ty * f_doppler * light_speed + 2 * delay
** 2 * v_ty * v_tz * light_speed ** 2) * math.sin(elevation) - 2
* f_doppler * h_r ** 2 * (v_ry - v_ty) - 2 * light_speed * (-2 *
v_tz * v_ty + v_rz * (v_ry - v_ty)) * delay * h_r + 2 *
f_doppler * light_speed ** 2 * v_ty * delay ** 2) * math.cos
(elevation) + (2 * f_doppler * (v_rz + v_tz) * h_r ** 2 - 2 *
```

```
light_speed * delay * (v_rx ** 2 + v_ry ** 2 - v_tz * v_rz -
v_tz ** 2 - f_doppler ** 2) * h_r + 2 * delay ** 2 * v_tz *
f_doppler * light_speed ** 2) * math.sin(elevation) + (v_rz ** 2
+ 2 * v_tz * v_rz + v_tz ** 2 + f_doppler ** 2) * h_r ** 2 + 2 *
light_speed * delay * f_doppler * (v_rz + 2 * v_tz) * h_r -
light_speed ** 2 * delay ** 2 * (v_rx ** 2 + v_ry ** 2 - v_tz **
2 - f_doppler ** 2)) * math.sin(elevation) ** 2 * v_rx ** 2) +
(h_r * (v_ry - v_ty) * math.cos(elevation) ** 3 + (-h_r * (v_rz
+ v_tz) * math.sin(elevation) - delay * v_tz * light_speed - h_r
* f_doppler) * math.cos(elevation) ** 2 - (v_ry - v_ty) * (delay
* light_speed * math.sin(elevation) + h_r) * math.cos(elevation)
+ (light_speed * f_doppler * delay + v_rz * h_r + h_r * v_tz) *
math.sin(elevation) + delay * v_tz * light_speed + h_r *
f_doppler) * v_rx ** 2) / ((v_ty ** 2 - v_tz ** 2) * math.cos
(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin(elevation) +
f_doppler) * math.cos(elevation) ** 3 + (2 * v_tz * math.sin
(elevation) * f_doppler + f_doppler ** 2 - v_rx ** 2 - 2 * v_ry
* v_ty + v_tz ** 2) * math.cos(elevation) ** 2 - 2 * v_ry *
(v_tz * math.sin(elevation) + f_doppler) * math.cos(elevation) +
v_rx ** 2 + v_ry ** 2) / math.sin(elevation) / v_rx
```

> $all\_y := allvalues(sol[2])$:

> $sol\_y\_1 := simplify(all\_y[1])$:

> $Python(sol\_y\_1)$

```
cg16 = y == (2 * math.cos(elevation) ** 4 * delay * v_ty * v_tz
* light_speed + (-light_speed * delay * (v_ty - v_tz) * (v_ty +
v_tz) * math.sin(elevation) + h_r * v_tz ** 2 + (2 * light_speed
* f_doppler * delay + v_rz * h_r) * v_tz + h_r * v_ty * (v_ry -
v_ty)) * math.cos(elevation) ** 3 + (((v_ry - 2 * v_ty) * h_r *
v_tz - v_ty * (2 * light_speed * f_doppler * delay + v_rz * h_r)
) * math.sin(elevation) - light_speed * delay * (v_ry + 2 *
v_ty) * v_tz + h_r * f_doppler * (v_ry - 2 * v_ty)) * math.cos
(elevation) ** 2 + ((-delay * v_tz ** 2 * light_speed - 2 * h_r
* v_tz * f_doppler - h_r * v_rz * f_doppler + light_speed *
delay * (v_rx ** 2 + v_ry * v_ty - f_doppler ** 2)) * math.sin
(elevation) - h_r * v_tz ** 2 + (-2 * light_speed * f_doppler *
delay - v_rz * h_r) * v_tz - h_r * (v_ry ** 2 - v_ry * v_ty +
f_doppler ** 2)) * math.cos(elevation) + v_ry * (light_speed *
f_doppler * delay + v_rz * h_r + h_r * v_tz) * math.sin
(elevation) + delay * v_ry * v_tz * light_speed + h_r * v_ry *
f_doppler + math.sqrt(-(-4 * math.cos(elevation) ** 3 * delay *
h_r * v_ty * v_tz * light_speed + (2 * h_r * light_speed * delay
* (v_ty - v_tz) * (v_ty + v_tz) * math.sin(elevation) + (v_tz +
v_ry - v_ty + v_rz) * (-v_tz + v_ry - v_ty - v_rz) * h_r ** 2 -
4 * delay * h_r * v_tz * f_doppler * light_speed + light_speed *
* 2 * delay ** 2 * (v_ty - v_tz) * (v_ty + v_tz)) * math.cos
(elevation) ** 2 + ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h_r **
2 + 4 * delay * h_r * v_ty * f_doppler * light_speed + 2 * delay
** 2 * v_ty * v_tz * light_speed ** 2) * math.sin(elevation) - 2
* f_doppler * h_r ** 2 * (v_ry - v_ty) - 2 * light_speed * (-2 *
v_tz * v_ty + v_rz * (v_ry - v_ty)) * delay * h_r + 2 *
f_doppler * light_speed ** 2 * v_ty * delay ** 2) * math.cos
(elevation) + (2 * f_doppler * (v_rz + v_tz) * h_r ** 2 - 2 *
light_speed * delay * (v_rx ** 2 + v_ry ** 2 - v_tz * v_rz -
v_tz ** 2 - f_doppler ** 2) * h_r + 2 * delay ** 2 * v_tz *
f_doppler * light_speed ** 2) * math.sin(elevation) + (v_rz ** 2
+ 2 * v_tz * v_rz + v_tz ** 2 + f_doppler ** 2) * h_r ** 2 + 2 *
```

```
light_speed * delay * f_doppler * (v_rz + 2 * v_tz) * h_r -
light_speed ** 2 * delay ** 2 * (v_rx ** 2 + v_ry ** 2 - v_tz **
2 - f_doppler ** 2)) * math.sin(elevation) ** 2 * v_rx ** 2)) /
((v_ty ** 2 - v_tz ** 2) * math.cos(elevation) ** 4 + 2 * v_ty *
(v_tz * math.sin(elevation) + f_doppler) * math.cos(elevation) *
* 3 + (2 * v_tz * math.sin(elevation) * f_doppler + f_doppler **
2 - v_rx ** 2 - 2 * v_ry * v_ty + v_tz ** 2) * math.cos
(elevation) ** 2 - 2 * v_ry * (v_tz * math.sin(elevation) +
f_doppler) * math.cos(elevation) + v_rx ** 2 + v_ry ** 2) /
math.sin(elevation)
```

```
> sol_y_2 := simplify(all_y[2]):
```

```
> Python(sol_y_2)
cg17 = y == (2 * math.cos(elevation) ** 4 * delay * v_ty * v_tz
* light_speed + (-light_speed * delay * (v_ty - v_tz) * (v_ty +
v_tz) * math.sin(elevation) + h_r * v_tz ** 2 + (2 * light_speed
* f_doppler * delay + v_rz * h_r) * v_tz + h_r * v_ty * (v_ry -
v_ty)) * math.cos(elevation) ** 3 + (((v_ry - 2 * v_ty) * h_r *
v_tz - v_ty * (2 * light_speed * f_doppler * delay + v_rz * h_r)
) * math.sin(elevation) - light_speed * delay * (v_ry + 2 *
v_ty) * v_tz + h_r * f_doppler * (v_ry - 2 * v_ty)) * math.cos
(elevation) ** 2 + ((-delay * v_tz ** 2 * light_speed - 2 * h_r
* v_tz * f_doppler - h_r * v_rz * f_doppler + light_speed *
delay * (v_rx ** 2 + v_ry * v_ty - f_doppler ** 2)) * math.sin
(elevation) - h_r * v_tz ** 2 + (-2 * light_speed * f_doppler *
delay - v_rz * h_r) * v_tz - h_r * (v_ry ** 2 - v_ry * v_ty +
f_doppler ** 2)) * math.cos(elevation) + v_ry * (light_speed *
f_doppler * delay + v_rz * h_r + h_r * v_tz) * math.sin
(elevation) + delay * v_ry * v_tz * light_speed + h_r * v_ry *
f_doppler - math.sqrt(-(-4 * math.cos(elevation) ** 3 * delay *
h_r * v_ty * v_tz * light_speed + (2 * h_r * light_speed * delay
* (v_ty - v_tz) * (v_ty + v_tz) * math.sin(elevation) + (v_tz +
v_ry - v_ty + v_rz) * (-v_tz + v_ry - v_ty - v_rz) * h_r ** 2 -
4 * delay * h_r * v_tz * f_doppler * light_speed + light_speed *
* 2 * delay ** 2 * (v_ty - v_tz) * (v_ty + v_tz)) * math.cos
(elevation) ** 2 + ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h_r **
2 + 4 * delay * h_r * v_ty * f_doppler * light_speed + 2 * delay
** 2 * v_ty * v_tz * light_speed ** 2) * math.sin(elevation) - 2
* f_doppler * h_r ** 2 * (v_ry - v_ty) - 2 * light_speed * (-2 *
v_tz * v_ty + v_rz * (v_ry - v_ty)) * delay * h_r + 2 *
f_doppler * light_speed ** 2 * v_ty * delay ** 2) * math.cos
(elevation) + (2 * f_doppler * (v_rz + v_tz) * h_r ** 2 - 2 *
light_speed * delay * (v_rx ** 2 + v_ry ** 2 - v_tz * v_rz -
v_tz ** 2 - f_doppler ** 2) * h_r + 2 * delay ** 2 * v_tz *
f_doppler * light_speed ** 2) * math.sin(elevation) + (v_rz ** 2
+ 2 * v_tz * v_rz + v_tz ** 2 + f_doppler ** 2) * h_r ** 2 + 2 *
light_speed * delay * f_doppler * (v_rz + 2 * v_tz) * h_r -
light_speed ** 2 * delay ** 2 * (v_rx ** 2 + v_ry ** 2 - v_tz **
2 - f_doppler ** 2)) * math.sin(elevation) ** 2 * v_rx ** 2)) /
((v_ty ** 2 - v_tz ** 2) * math.cos(elevation) ** 4 + 2 * v_ty *
(v_tz * math.sin(elevation) + f_doppler) * math.cos(elevation) *
* 3 + (2 * v_tz * math.sin(elevation) * f_doppler + f_doppler **
2 - v_rx ** 2 - 2 * v_ry * v_ty + v_tz ** 2) * math.cos
(elevation) ** 2 - 2 * v_ry * (v_tz * math.sin(elevation) +
f_doppler) * math.cos(elevation) + v_rx ** 2 + v_ry ** 2) /
math.sin(elevation)
```

```
> sol_x_1 := simplify(all_x[1]):
```

> $simplify\left(eval\left(all\_x[1], \left\{ elevation = 0.91943,\ v\_ty = 1183.799,\ v\_tz = -671.829,\right.\right.\right.$

$v\_rx = 6043.852,\ v\_ry = -4654.310,\ v\_rz = -315.129,\ h\_r = 612487.690,$

$\left.\left.\left. f\_doppler = -2500.0,\ delay = \frac{2}{1.023 \cdot 10^6},\ light\_speed = 299792458.0 \right\} \right) \right)$

 # Check solution

$$x = 29781.10518 \quad (4)$$

> $simplify\left(eval\left(all\_x[2], \left\{ elevation = 0.91943,\ v\_ty = 1183.799,\ v\_tz = -671.829,\right.\right.\right.$

$v\_rx = 6043.852,\ v\_ry = -4654.310,\ v\_rz = -315.129,\ h\_r = 612487.690,$

$\left.\left.\left. f\_doppler = -2500.0,\ delay = \frac{2}{1.023 \cdot 10^6},\ light\_speed = 299792458.0 \right\} \right) \right)$

 # Check solution

$$x = 14856.48351 \quad (5)$$

> $simplify\left(eval\left(all\_y[1], \left\{ elevation = 0.91943,\ v\_ty = 1183.799,\ v\_tz = -671.829,\right.\right.\right.$

$v\_rx = 6043.852,\ v\_ry = -4654.310,\ v\_rz = -315.129,\ h\_r = 612487.690,$

$\left.\left.\left. f\_doppler = -2500.0,\ delay = \frac{2}{1.023 \cdot 10^6},\ light\_speed = 299792458.0 \right\} \right) \right)$

 # Check solution

$$y = -4536.145464 \quad (6)$$

> $simplify\left(eval\left(all\_y[2], \left\{ elevation = 0.91943,\ v\_ty = 1183.799,\ v\_tz = -671.829,\right.\right.\right.$

$v\_rx = 6043.852,\ v\_ry = -4654.310,\ v\_rz = -315.129,\ h\_r = 612487.690,$

$\left.\left.\left. f\_doppler = -2500.0,\ delay = \frac{2}{1.023 \cdot 10^6},\ light\_speed = 299792458.0 \right\} \right) \right)$

 # Check solution

$$y = -32292.08120 \quad (7)$$

> *Python(eq1)*

```
cg18 = delay == 0.1e1 / light_speed * (math.sqrt(x ** 2 + (y +
h_r / math.tan(elevation)) ** 2 + h_r ** 2) - h_r / math.sin
(elevation) - y * math.cos(elevation))
```

> *Python(eq2)*

```
cg19 = f_doppler == -v_ty * math.cos(elevation) - v_tz * math.
sin(elevation) + (v_rx * x + v_ry * (y + h_r / math.tan
(elevation)) - v_rz * h_r) * (x ** 2 + (y + h_r / math.tan
(elevation)) ** 2 + h_r ** 2) ** (-0.1e1 / 0.2e1)
```

>