>
$$eq1 := delay = \sqrt{x^2 + \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2} - \frac{h}{\sin(elevation)} - y$$

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

$$- y\cos(elevation)$$
> $eq2 := f_-d - v_-ty \cdot \cos(elevation) - v_-tz \cdot \sin(elevation)$

$$+ \frac{v_-rx \cdot x + v_-ry \cdot \left(y + \frac{h}{\tan(elevation)}\right) - v_-rz \cdot h}{\sqrt{x^2 + \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2}}$$

$$eq2 := f_-d - v_-ty \cdot \cos(elevation) - v_-tz \cdot \sin(elevation)$$

$$+ \frac{v_-rx \cdot x + v_-ry \cdot \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2}{\sqrt{x^2 + \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2}}$$
> $sys := \{eq1, eq2\}$

$$sys := \{eq1, eq2\}$$

$$sys := \{eq1, eq2\}$$

$$sys := \{delay = \sqrt{x^2 + \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2} - \frac{h}{\sin(elevation)}$$

$$- y \cos(elevation), f_-d = -v_-ty \cos(elevation) - v_-tz \cdot \sin(elevation)$$

$$+ \frac{v_-rx \cdot x + v_-ry \cdot \left(y + \frac{h}{\tan(elevation)}\right) - v_-rz \cdot h}{\sqrt{x^2 + \left(y + \frac{h}{\tan(elevation)}\right)^2 + h^2}}$$

$$\Rightarrow sol := solve(sys, (x, y)):$$

$$\Rightarrow sol_x := solve(sys, (x, y)):$$

$$\Rightarrow sol_x := simplify(all_x[1]):$$

$$sol_x x := simplify(all_x[1]):$$

$$sol_x x := simplify(all_x[1]):$$

$$sol_x x := simplify(all_x[1]):$$

$$- f_-d) \cos(elevation) + v_-ry \cdot \left(-v_-tz \cdot \sin(elevation) - v_-tz \cdot \sin(elevation) - f_-d) \cos(elevation) + v_-ry \cdot \left(-v_-tz \cdot \sin(elevation)^3 \cdot delay \cdot hv_-tz + delay^2 \cdot \left(v_-ty^2 - v_-tz^2\right)\right)$$

```
\cos(elevation)^{2} + \left(\left(-2\left(v_{-}rz + v_{-}tz\right)\left(v_{-}ry - v_{-}ty\right)h^{2} + 4 \ delay f_{-}dh \ v_{-}ty\right) + 2 \ delay^{2} \ v_{-}ty \ v_{-}tz\right) \sin(elevation) - 2 \ h^{2} \ f_{-}d \left(v_{-}ry - v_{-}ty\right) - 2 \ \left(-2 \ v_{-}ty \ v_{-}tz\right) + v_{-}rz \ \left(v_{-}ry - v_{-}ty\right)\right) delay \ h + 2 \ f_{-}d \ v_{-}ty \ delay^{2}\right) \cos(elevation) + \left(2 \ f_{-}d \left(v_{-}rz + v_{-}tz\right)h^{2} + 2 \ delay \left(f_{-}d^{2} - v_{-}rx^{2} - v_{-}ry^{2} + v_{-}rz \ v_{-}tz\right) + v_{-}tz^{2}\right) h + 2 \ f_{-}d \ delay^{2} \ v_{-}tz\right) \sin(elevation) + \left(f_{-}d^{2} + v_{-}rz^{2} + 2 \ v_{-}rz \ v_{-}tz\right) + v_{-}tz^{2}\right) h^{2} + 2 \ f_{-}d \ delay \ \left(v_{-}rz + 2 \ v_{-}tz\right)h + delay^{2} \ \left(f_{-}d^{2} - v_{-}rx^{2} - v_{-}ry^{2} + v_{-}rz^{2}\right) + v_{-}tz^{2}\right) \sin(elevation)^{2}\right)^{1/2} + v_{-}rx^{2} \left(h \left(v_{-}ry - v_{-}ty\right)\cos(elevation)^{3} + \left(-h \left(v_{-}rz + v_{-}tz\right)\sin(elevation) - f_{-}dh - delay \ v_{-}tz\right)\cos(elevation)^{2} - \left(v_{-}ry - v_{-}ty\right) \left(delay \sin(elevation) + f_{-}dh - delay \ v_{-}tz\right) \cos(elevation) \left(\left(v_{-}ty^{2} - v_{-}tz^{2}\right)\cos(elevation)^{4} + 2 \ v_{-}tz\right) \left(v_{-}tz\sin(elevation) + f_{-}d\right)\cos(elevation)^{3} + \left(2 \ f_{-}d \ v_{-}tz\sin(elevation) + f_{-}d^{2} - v_{-}rx^{2} - 2 \ v_{-}ry \ v_{-}ty\right) + v_{-}tz^{2}\right)\cos(elevation)^{2} - 2 \ v_{-}ry \ \left(v_{-}tz\sin(elevation) + f_{-}d\right)\cos(elevation) + v_{-}rx^{2} + v_{-}ry^{2}\right)\right)
```

> with(CodeGeneration):

> Python(sol_x_1)
cg = x == ((-math.cos(elevation) ** 2 * v_ty + (-v_tz * math.sin (elevation) - f_d) * math.cos(elevation) + v_ry) * math.sqrt(-v_rx ** 2 * (-4 * math.cos(elevation) ** 3 * delay * h * v_ty * v_tz + (2 * h * 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 ** 2 - 4 * delay * f_d * h * v_tz + delay ** 2 * (v_ty ** 2 - v_tz ** 2)) * math.cos(elevation) ** 2 + ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h ** 2 + 4 * delay * f_d * h * v_tz + delay * f_d * h * v_ty + 2 * delay ** 2 * v_ty * v_tz) * math.sin(elevation) - 2 * h ** 2 * f_d * (v_ry - v_ty) * 2 * (-2 * v_ty * v_tz + v_rz * (v_ry - v_ty)) * delay * h + 2 * f_d * v_ty * delay ** 2) * math.cos(elevation) + (2 * f_d * (v_rz + v_tz) * h ** 2 + 2 * delay * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_rz * v_tz + v_tz * 2) * h + 2 * f_d * delay ** 2 * v_tz) * math.sin(elevation) + (f_d ** 2 + v_rz ** 2 + 2 * v_rz * v_tz + v_tz * 2 * (f_d ** 2 + v_rz ** 2 + 2 * v_tz) * math.sin(elevation) ** 2 + 2 * f_d * delay * (v_rz + 2 * v_tz) * h + delay ** 2 * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_tz * 2) * h ** 2 + 2 * v_tz * math.sin(elevation) ** 2 - v_rx ** 2 - v_rx ** 2 - v_ry ** 2 + v_tz * 2) * h * delay * (v_rz + v_tz) * math.sin(elevation) ** 2 - v_rx ** 2 - v_rx ** 2 - v_ry ** 2 + v_tz * 2) * h * delay * v_tz) * math.sin(elevation) ** 2 - (v_ry - v_ty) * (delay * math.sin(elevation) + f_d * h - delay * v_tz) * math.sin(elevation) ** 2 - (v_ry - v_ty) * (delay * math.sin(elevation) + f_d * h + delay * v_tz) * math.sin(elevation) ** 2 - (v_ry - v_ty) * (delay * v_tz) * math.sin(elevation) + f_d * h + delay * v_tz) * math.sin(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin (elevation) + f_d * h + delay * v_tz) * math.sin(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin (elevation) + f_d * v_tz * math.sin (elevation) + f_d * v_tz * v_ty * v_tz * math.sin (elevation) + f_d * v_tz * v_ty * v_tz * v_ty * v_ty * v_tz * v_ty * v

```
v_tz ** 2) * math.cos(elevation) ** 2 - 2 * v_ry * (v_tz * math.
  z_1 = z_2 = z_3 
  \rightarrow sol_x_2 := simplify(all_x[2])
  sol_x_2 := x = ((\cos(elevation)^2 v_ty + (v_tz\sin(elevation) + f_d)\cos(elevation))
               -v_ry
               (-v_rx^2)(-4\cos(elevation)^3) delayhv_tyv_tz + (2hdelay(v_ty-v_tz)(v_ty+v_tz)\sin(elev_tz))
               -v_{ty}-v_{rz}) h^2-4 delay f_{t}dhv_{tz}+delay^2(v_{ty}^2-v_{tz}^2))
               \cos(elevation)^2 + ((-2(v_rz + v_tz)(v_ry - v_ty))h^2 + 4 delay f_d h v_ty)
               +2 delay^2 v_t v_t z \sin(elevation) - 2 h^2 f_d (v_r v_t v_t v_t z)
               +v_rz(v_ry-v_ty) delay h+2f_dv_ty delay<sup>2</sup> cos(elevation)
               + (2 f_{-}d (v_{-}rz + v_{-}tz) h^{2} + 2 delay (f_{-}d^{2} - v_{-}rx^{2} - v_{-}ry^{2} + v_{-}rzv_{-}tz)
               +v_{t}z^{2}) h+2f_{d}delay^{2}v_{t}z) sin(elevation) + (f_{d}^{2}+v_{r}z^{2}+2v_{r}zv_{t}z)
               +v_{t}z^{2}) h^{2} + 2 f_{d} delay (v_{r}z + 2 v_{t}z) h + delay^{2} (f_{d}^{2} - v_{r}x^{2} - v_{r}y^{2})
               +v_{-}tz^{2}) \sin(elevation)^{2} +v_{-}rx^{2}(h(v_{-}ry-v_{-}ty)\cos(elevation)^{3}+(v_{-}tz^{2}))
              -h(v_rz + v_tz)\sin(elevation) - f_dh - delayv_tz)\cos(elevation)^2 - (v_ry)
               -v_{ty}) (delay \sin(elevation) + h) \cos(elevation) + (delay <math>f_{ty} + v_{ty} + h
               +hv_{t}z \sin(elevation) + f_{d}h + delayv_{t}z) / (v_{r}x\sin(elevation)) ((v_{t}y^{2}))
               -v_{t}z^{2} cos(elevation)<sup>4</sup> + 2 v_{t}y (v_{t}z\sin(elevation) + f_{d}) cos(elevation)<sup>3</sup>
               + (2 f_{-} dv_{-} tz \sin(elevation) + f_{-} d^{2} - v_{-} rx^{2} - 2 v_{-} ry v_{-} ty
               +v_{t}z^{2} cos(elevation)<sup>2</sup> – 2 v_{t}y (v_{t}z\sin(elevation) + f_{d}) cos(elevation)
               + v_{r}x^{2} + v_{r}v^{2})
> Python(sol_x_2)
cg0 = x == ((math.cos(elevation) ** 2 * v_ty + (v_tz * math.sin
(elevation) + f_d) * math.cos(elevation) - v_ry) * math.sqrt(-
v_rx ** 2 * (-4 * math.cos(elevation) ** 3 * delay * h * v_ty *
v_tz + (2 * h * 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 ** 2 - 4 * delay * f_d * h * v_tz + delay ** 2 *
(v_ty ** 2 - v_tz ** 2)) * math.cos(elevation) ** 2 + ((-2 *
(v_rz + v_tz) * (v_ry - v_ty) * h ** 2 + 4 * delay * f_d * h *
v_ty + 2 * delay ** 2 * v_ty * v_tz) * math.sin(elevation) - 2 *
h ** 2 * f_d * (v_ry - v_ty) - 2 * (-2 * v_ty * v_tz + v_rz *
(v_ry - v_ty)) * delay * h + 2 * f_d * v_ty * delay ** 2) *
math.cos(elevation) + (2 * f_d * (v_rz + v_tz) * h ** 2 + 2 *
delay * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_rz * v_tz + v_tz *
2) * h + 2 * f_d * delay ** 2 * v_tz) * math.sin(elevation) +
(f_d ** 2 + v_rz ** 2 + 2 * v_rz * v_tz + v_tz ** 2) * h ** 2 +
```

```
2 * f_d * delay * (v_rz + 2 * v_tz) * h + delay ** 2 * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_tz ** 2)) * math.sin(elevation) ** 2) + v_rx ** 2 * (h * (v_ry - v_ty) * math.cos(elevation) ** 3 + (-h * (v_rz + v_tz) * math.sin(elevation) - f_d * h - delay * v_tz) * math.cos(elevation) ** 2 - (v_ry - v_ty) * (delay * math.sin(elevation) + h) * math.cos(elevation) + (delay * f_d + v_rz * h + h * v_tz) * math.sin(elevation) + f_d * h + delay * v_tz)) / v_rx / math.sin(elevation) / ((v_ty ** 2 - v_tz ** 2) * math.cos(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin (elevation) + f_d) * math.cos(elevation) ** 3 + (2 * f_d * v_tz * math.sin(elevation) + f_d ** 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_d) * math.cos(elevation) + v_rx ** 2 + v_ry * * 2)
      2)
\rightarrow all_y := allvalues(sol[2]):
 > sol_y_1 := simplify(all_y[1])
 sol_{y_1} := y = (2 \cos(elevation)^4 delay v_t y_t z + ((-delay v_t y^2 + ((-delay v_t y^
              + delay v_t z^2 \sin(elevation) + h v_t z^2 + (2 delay f_d + v_r z h) v_t z
             +hv_t(v_ry-v_ty) cos(elevation)<sup>3</sup> + (((v_ry-2v_ty) hv_tz
              -v_ty(2 delay f_d + v_rz h)) \sin(elevation) - delay(v_ry + 2 v_ty) v_tz
             +hf_{-}d(v_{-}ry-2v_{-}ty))\cos(elevation)^{2}+((-delayv_{-}tz^{2}-2f_{-}dhv_{-}tz)^{2})
             -f_dhv_rz - delay(f_d^2 - v_rx^2 - v_ryv_ty))\sin(elevation) - hv_tz^2 + (elevation)
             -2 \ delay f_d - v_rzh) v_tz - h (f_d^2 + v_ry^2 - v_ry v_ty)) \cos(elevation)
             +v_ry (delay f_d+v_rzh+hv_tz) \sin(elevation)+f_dhv_ry
              + delay v_ry v_tz
              -(-v_rx^2)(-4\cos(elevation)^3) delay hv_tyv_tz + (2hdelay(v_ty-v_tz)(v_ty+v_tz)\sin(elevation)^3)
             +2 delay^2 v_t v_t z \sin(elevation) - 2 h^2 f_d (v_r y_t v_t z) - 2 (-2 v_t y_t z)
             +v_rz(v_ry-v_ty) delay h+2f_dv_ty delay<sup>2</sup>) cos(elevation)
             + (2 f_{-}d (v_{-}rz + v_{-}tz) h^{2} + 2 delay (f_{-}d^{2} - v_{-}rx^{2} - v_{-}ry^{2} + v_{-}rzv_{-}tz)
              +v_{t}z^{2}) h+2 f_{d} delay^{2} v_{t}z) sin(elevation) + (f_{d}^{2} + v_{r}z^{2} + 2 v_{r}z v_{t}z)
             +v_{z}^{2} h^{2} + 2 f_{z} d d e lay (v_{r}^{2} + 2 v_{z}^{2}) h + d e lay^{2} (f_{z} d^{2} - v_{r}^{2} v_{z}^{2} - v_{z}^{2})
              +v_{t}z^{2}) \sin(elevation)^{2} /(\sin(elevation)(v_{t}y^{2}))
              -v_{t}z^{2} cos(elevation)<sup>4</sup> + 2 v_{t}y (v_{t}z\sin(elevation) + f_{d}) cos(elevation)<sup>3</sup>
             + (2 f_{-} dv_{-} tz \sin(elevation) + f_{-} d^{2} - v_{-} rx^{2} - 2 v_{-} ry v_{-} ty
             +v_{-}tz^{2}) \cos(elevation)^{2} - 2v_{-}ry(v_{-}tz\sin(elevation) + f_{-}d)\cos(elevation)
              + v_{r}x^{2} + v_{r}v^{2})
```

```
> Python(sol_y_1)
cg1 = y == (2 * math.cos(elevation) ** 4 * delay * v_ty * v_tz +
((-delay * v_ty ** 2 + delay * v_tz ** 2) * math.sin(elevation)
+ h * v_tz ** 2 + (2 * delay * f_d + v_rz * h) * v_tz + h * v_ty
* (v_ry - v_ty)) * math.cos(elevation) ** 3 + (((v_ry - 2 *
v_ty) * h * v_tz - v_ty * (2 * delay * f_d + v_rz * h)) * math.
sin(elevation) - delay * (v_ry + 2 * v_ty) * v_tz + h * f_d *
(v_ry - 2 * v_ty)) * math.cos(elevation) ** 2 + ((-delay * v_tz
** 2 - 2 * f_d * h * v_tz - f_d * h * v_rz - delay * (f_d ** 2 -
v_rx ** 2 - v_ry * v_ty)) * math.sin(elevation) - h * v_tz ** 2
+ (-2 * delay * f_d - v_rz * h) * v_tz - h * (f_d ** 2 + v_ry ** 2 - v_rv * v_ty)) * math.cos(elevation) + v_ry * (delay * f_d +
 2 - v_ry * v_ty)) * math.cos(elevation) + v_ry * (delay * f_d + v_rz * h + h * v_tz) * math.sin(elevation) + f_d * h * v_ry + delay * v_ry * v_tz - math.sqrt(-v_rx ** 2 * (-4 * math.cos (elevation) ** 3 * delay * h * v_ty * v_tz + (2 * h * delay *
(elevation) ** 3 * delay * h * v_ty * v_tz + (2 * h * 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 ** 2 - 4 * delay * f_d * h * v_tz + delay ** 2 * (v_ty ** 2 - v_tz ** 2)) * math.cos(elevation) ** 2 + ((-2 * (v_rz + v_tz) * (v_ry - v_ty) * h ** 2 + 4 * delay * f_d * h * v_ty + 2 * delay ** 2 * v_ty * v_tz) * math.sin(elevation) - 2 * h ** 2 * f_d * (v_ry - v_ty) - 2 * (-2 * v_ty * v_tz + v_rz * (v_ry - v_ty)) * delay * h + 2 * f_d * v_ty * delay ** 2) * math.cos(elevation) + (2 * f_d * (v_rz + v_tz) * h ** 2 + 2 * delay * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_rz * v_tz + v_tz ** 2) * h + 2 * f_d * delay ** 2 * v_tz) * math.sin(elevation) + (f_d ** 2 + v_rz ** 2 + 2 * v_rz * v_tz + v_tz ** 2) * h ** 2 + 2 * f_d * delay * (v_rz + 2 * v_tz) * h + delay ** 2 * (f_d ** 2 - v_rx ** 2 - v_ry ** 2 + v_tz ** 2)) * math.sin(elevation) / ((v_ty ** 2 - v_tz ** 2) * math.cos(elevation) ** 4 + 2 * v_ty *
  * (v_tz * math.sin(elevation) + f_d) * math.cos(elevation) +
  _v _`r x ** 2 + v _ ry *`* 2)
  > sol_y_2 := simplify(all_y[2])
  sol_{y_2} := y = (2 \cos(elevation)^4 delay v_t y_t z + ((-delay v_t y^2 + ((-delay v_t y^
                   + delay v_t z^2 \sin(elevation) + h v_t z^2 + (2 delay f_d + v_r z h) v_t z
                   + h v_{ty} (v_{ry} - v_{ty}) \cos(elevation)^{3} + (((v_{ry} - 2 v_{ty}) h v_{tz})
                   -v_t y (2 delay f_d + v_r z h)) \sin(elevation) - delay (v_r y + 2 v_t y) v_t z
                   + h f_d (v_r y - 2 v_t y) \cos(elevation)^2 + ((-delay v_t z^2 - 2 f_d h v_t z))
                   -f_{-}dhv_{-}rz - delay(f_{-}d^{2} - v_{-}rx^{2} - v_{-}ryv_{-}ty)) \sin(elevation) - hv_{-}tz^{2} + (elevation)
                  -2 \ delay f_d - v_rzh) v_tz - h (f_d^2 + v_rv^2 - v_ryv_ty)) \cos(elevation)
                   +v_ry (delay f_d+v_rzh+hv_tz) \sin(elevation)+f_dhv_ry
                   + delay v_ry v_tz
```

```
+2 delay^{2} v_{ty} v_{tz} \sin(elevation) - 2 h^{2} f_{d} (v_{ry} - v_{ty}) - 2 (-2 v_{ty} v_{tz})
     +v_rz(v_ry-v_ty) delay h+2f_dv_ty delay l) cos(elevation)
     + (2 f_{-}d (v_{rz} + v_{tz}) h^2 + 2 delay (f_{-}d^2 - v_{rx}^2 - v_{ry}^2 + v_{rz}v_{tz})
     +v_{t}z^{2}) h + 2 f_{d} delay^{2} v_{t}z) sin(elevation) + (f_{d}^{2} + v_{r}z^{2} + 2 v_{r}zv_{t}z)
     +v_{-}tz^{2}) h^{2} + 2 f_{-}d delay (v_{-}rz + 2 v_{-}tz) h + delay^{2} (f_{-}d^{2} - v_{-}rx^{2} - v_{-}ry^{2})
     +v_{t}z^{2}) \sin(elevation)^{2} /(\sin(elevation)(v_{t}y^{2})
     -v_{t}z^{2} cos(elevation)<sup>4</sup> + 2 v_{t}y (v_{t}z sin(elevation) + f_{d}) cos(elevation)<sup>3</sup>
     + (2 f_d v_t z \sin(elevation) + f_d^2 - v_r x^2 - 2 v_r y v_t y)
     +v_{t}z^{2} \cos(elevation)^{2} - 2v_{r}y(v_{t}z\sin(elevation) + f_{d})\cos(elevation)
+ v_{r}x^{2} + v_{r}y^{2})
((v_ty ** 2 - v_tz ** 2) * math.cos(elevation) ** 4 + 2 * v_ty * (v_tz * math.sin(elevation) + f_d) * math.cos(elevation) ** 3 + (2 * f_d * v_tz * math.sin(elevation) + f_d ** 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_d) * math.cos(elevation) +
v_rx ** 2 + v_ry ** 2)
```