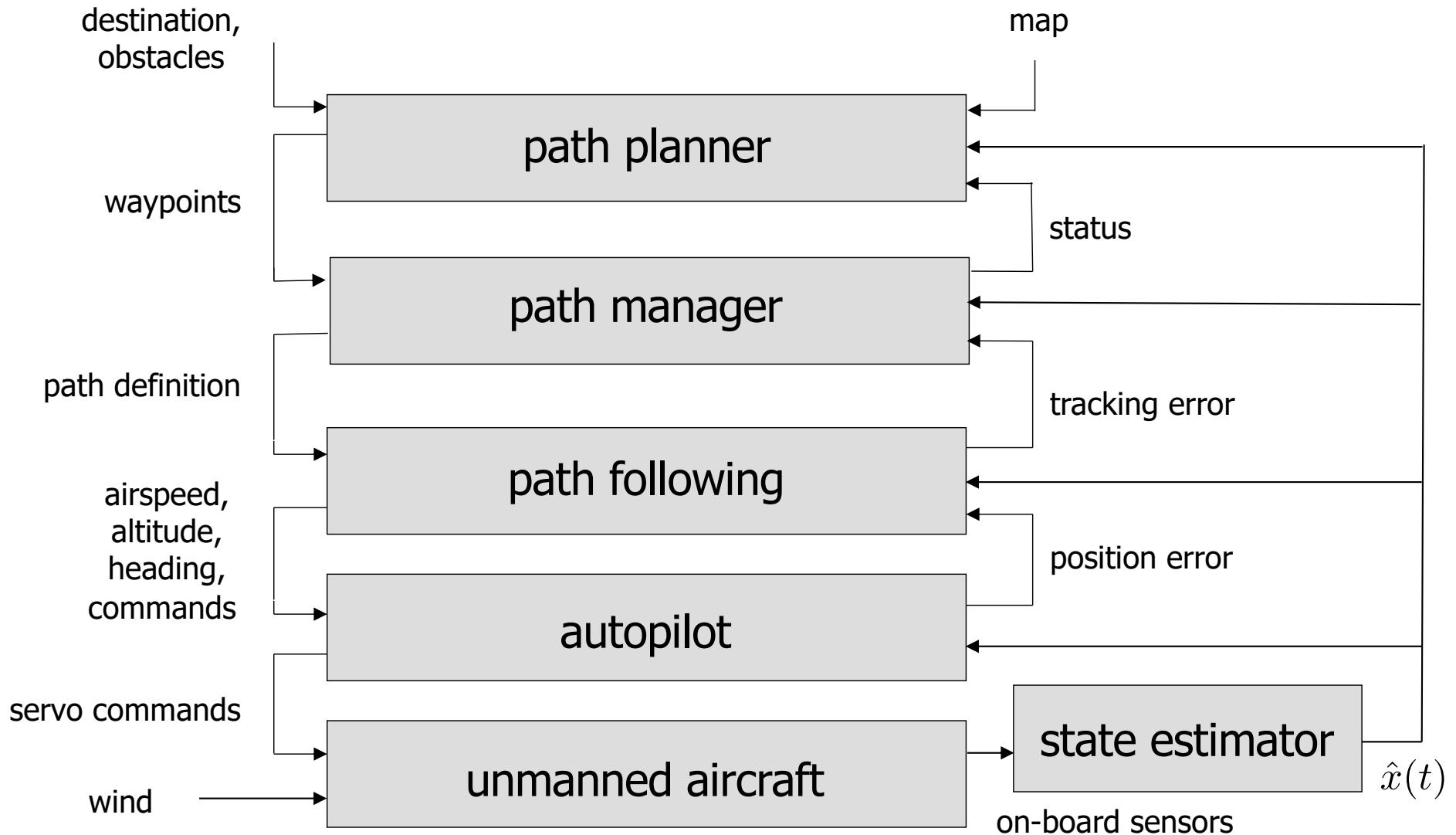




Chapter 10

Path Following

Control Architecture



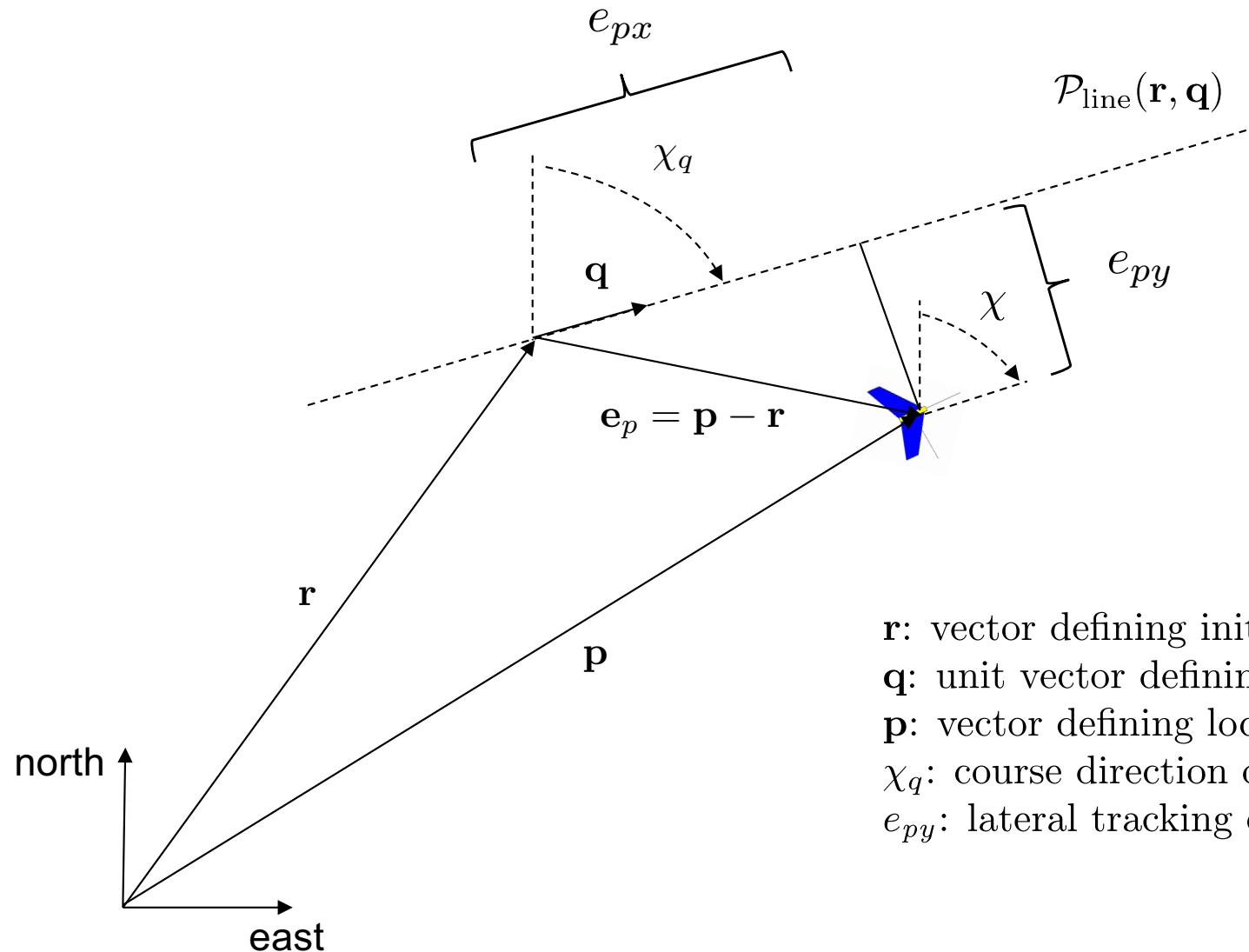
Path Following

- For small UAVs, a major issue is wind
 - Always present to some degree
 - Usually significant with respect to commanded airspeed
- Wind makes traditional trajectory tracking approaches difficult, if not infeasible
 - Have to know the wind precisely at every instant to determine desired airspeed
- Better approach: path following
- Rather than “follow this trajectory”, we control UAV to “stay on this path”

Path Types

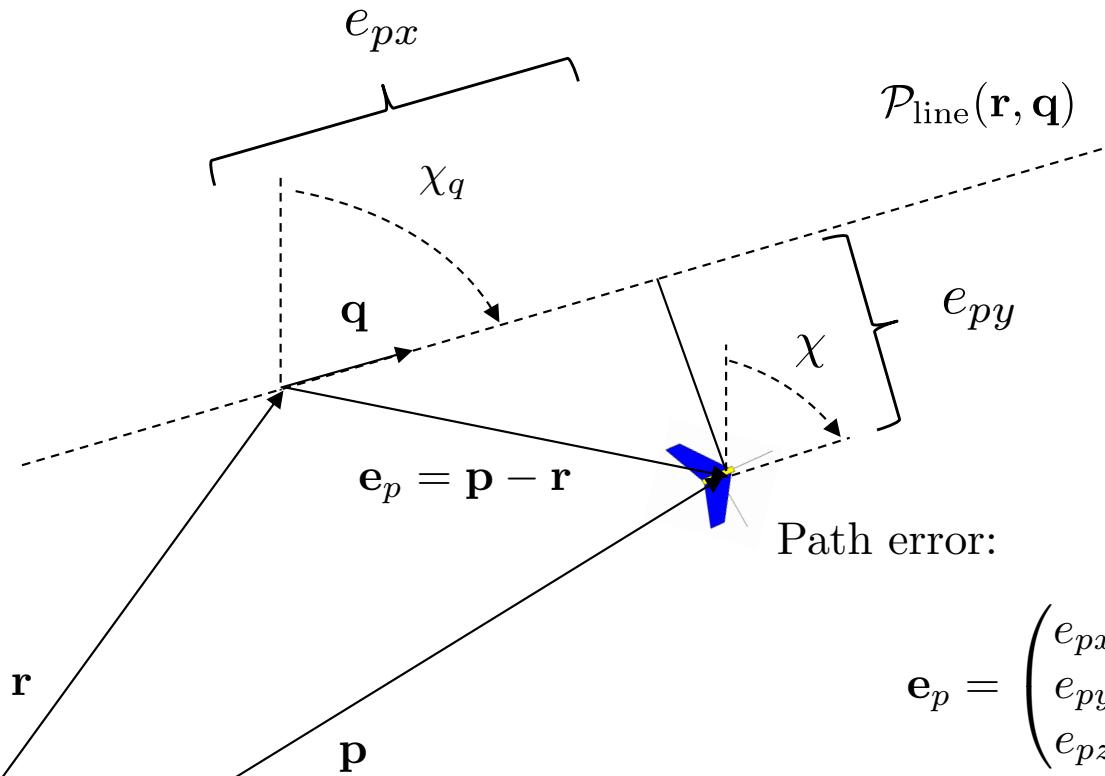
- We will focus on two types of paths to follow:
 - Straight lines between two points in 3-D
 - Inclination of path within climb capabilities of UAV
 - Circular orbits or arcs in the horizontal plane
- Paths for common applications can be built up from these path primitives
 - Methods for following other types of paths found in literature

Straight Line Path Description



\mathbf{r} : vector defining initiation of path
 \mathbf{q} : unit vector defining direction of path
 \mathbf{p} : vector defining location of MAV
 χ_q : course direction of path
 e_{py} : lateral tracking error

Lateral Tracking Problem

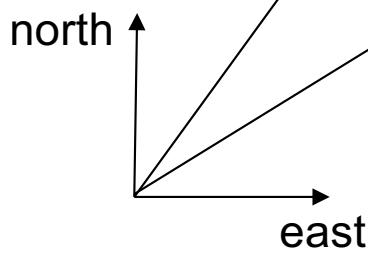


Path error:

$$\mathbf{e}_p = \begin{pmatrix} e_{px} \\ e_{py} \\ e_{pz} \end{pmatrix} \triangleq \mathcal{R}_i^{\mathcal{P}} (\mathbf{p}^i - \mathbf{r}^i)$$

where the transformation from inertial frame to path frame is

$$\mathcal{R}_i^{\mathcal{P}} \triangleq \begin{pmatrix} \cos \chi_q & \sin \chi_q & 0 \\ -\sin \chi_q & \cos \chi_q & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



Lateral Tracking Problem

Relative error dynamics in path frame:

$$\begin{pmatrix} \dot{e}_{px} \\ \dot{e}_{py} \end{pmatrix} = \begin{pmatrix} \cos \chi_q & \sin \chi_q \\ -\sin \chi_q & \cos \chi_q \end{pmatrix} \begin{pmatrix} V_g \cos \chi \\ V_g \sin \chi \end{pmatrix}$$

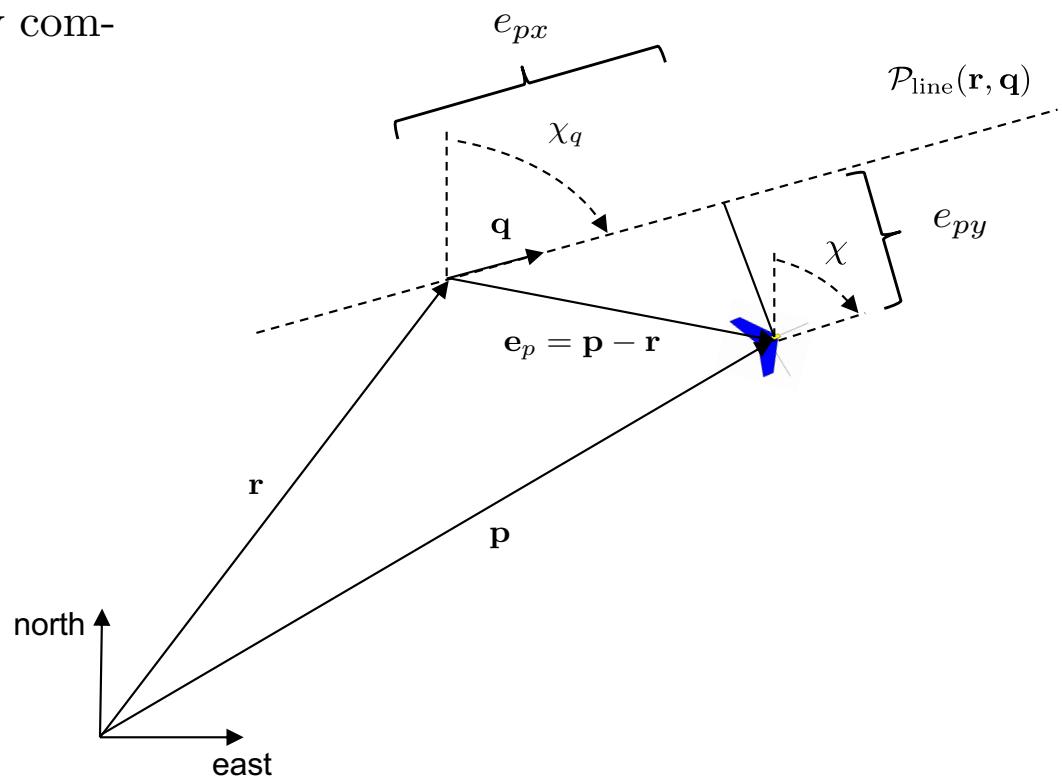
$$= V_g \begin{pmatrix} \cos(\chi - \chi_q) \\ \sin(\chi - \chi_q) \end{pmatrix}$$

Regulate the cross-track error e_{py} to zero by commanding the course angle:

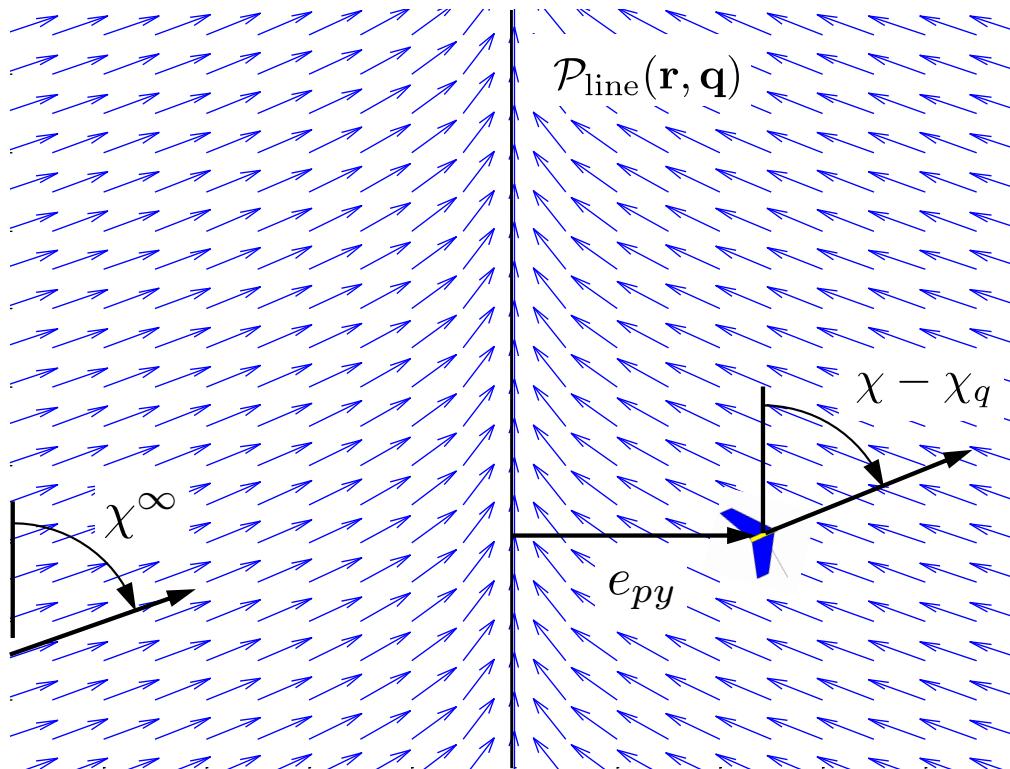
$$\dot{e}_{py} = V_g \sin(\chi - \chi_q)$$

$$\ddot{\chi} = b_{\dot{\chi}}(\dot{\chi}^c - \dot{\chi}) + b_\chi(\chi^c - \chi)$$

Select χ^c so that $e_{py} \rightarrow 0$



Lateral Tracking - Vector Field Concept

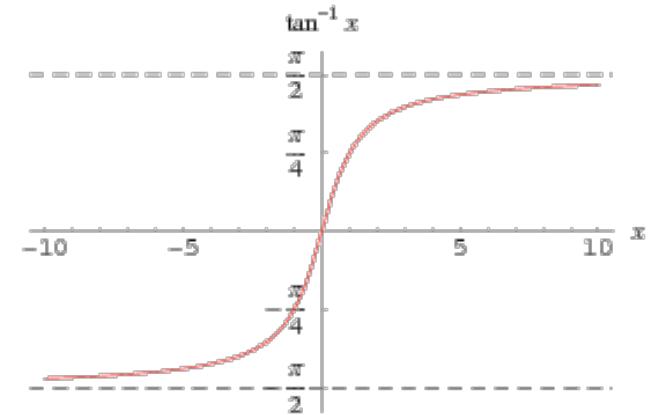


For a general line, the commanded course is

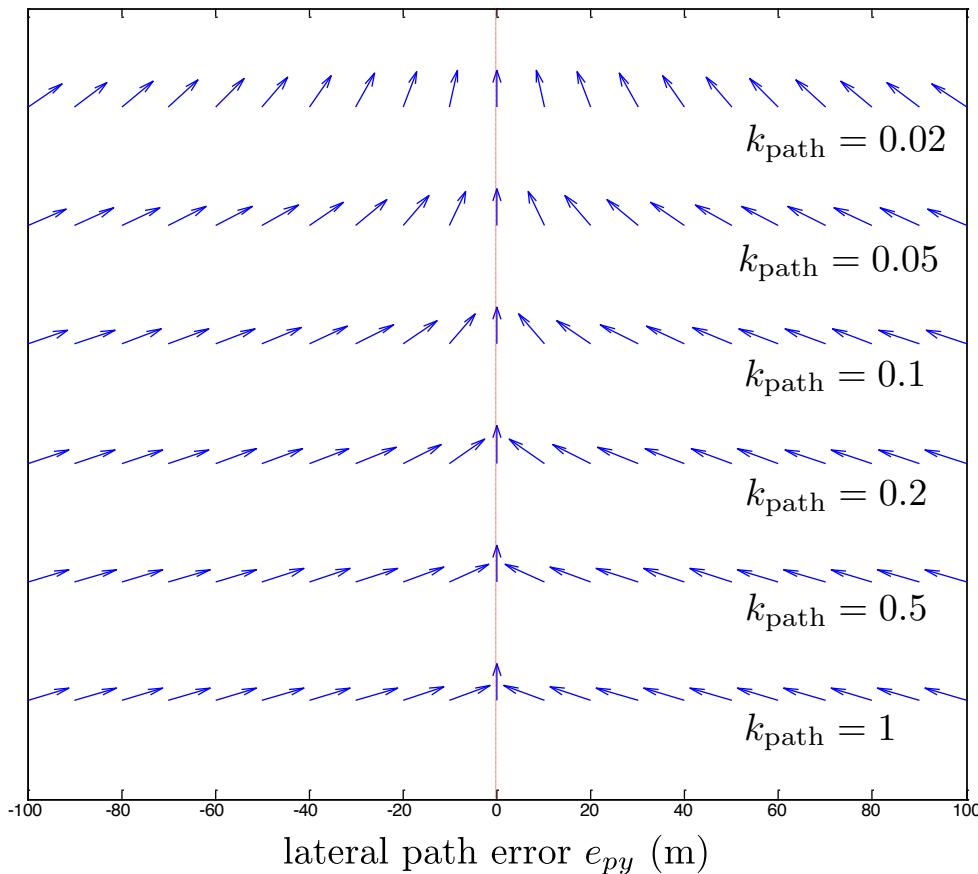
$$\chi^c = \chi_q - \chi^\infty \frac{2}{\pi} \tan^{-1}(k_{\text{path}} e_{py})$$

Desired course based on cross-track error:

$$\chi_d(e_{py}) = -\chi^\infty \frac{2}{\pi} \tan^{-1}(k_{\text{path}} e_{py})$$



Vector Field Tuning



k_{path} is a positive constant that affects the rate of transition of the desired course

- k_{path} large \rightarrow short, abrupt transition
- k_{path} small \rightarrow long, gradual transition

Rule of thumb:

$$k_{\text{path}} \approx \frac{1}{R_{\min}},$$

where R_{\min} is the minimum turn radius of the aircraft.

Lyapunov's 2nd Method

For a system having a state vector x , consider an energy-like (Lyapunov) function $V(x) : \mathbb{R}^n \mapsto \mathbb{R}$ such that

$$V(x) > 0, \forall x \neq 0 \text{ (positive definite)}$$

$$V(0) = 0$$

and

$$\dot{V}(x) < 0, \forall x \neq 0 \text{ (negative definite)}$$

$$\dot{V}(0) = 0.$$

If such a function $V(x)$ can be defined, then x goes to zero asymptotically and the system is stable.

Lateral Tracking Stability Analysis

Define the Lyapunov function $W(e_{py}) = \frac{1}{2}e_{py}^2$

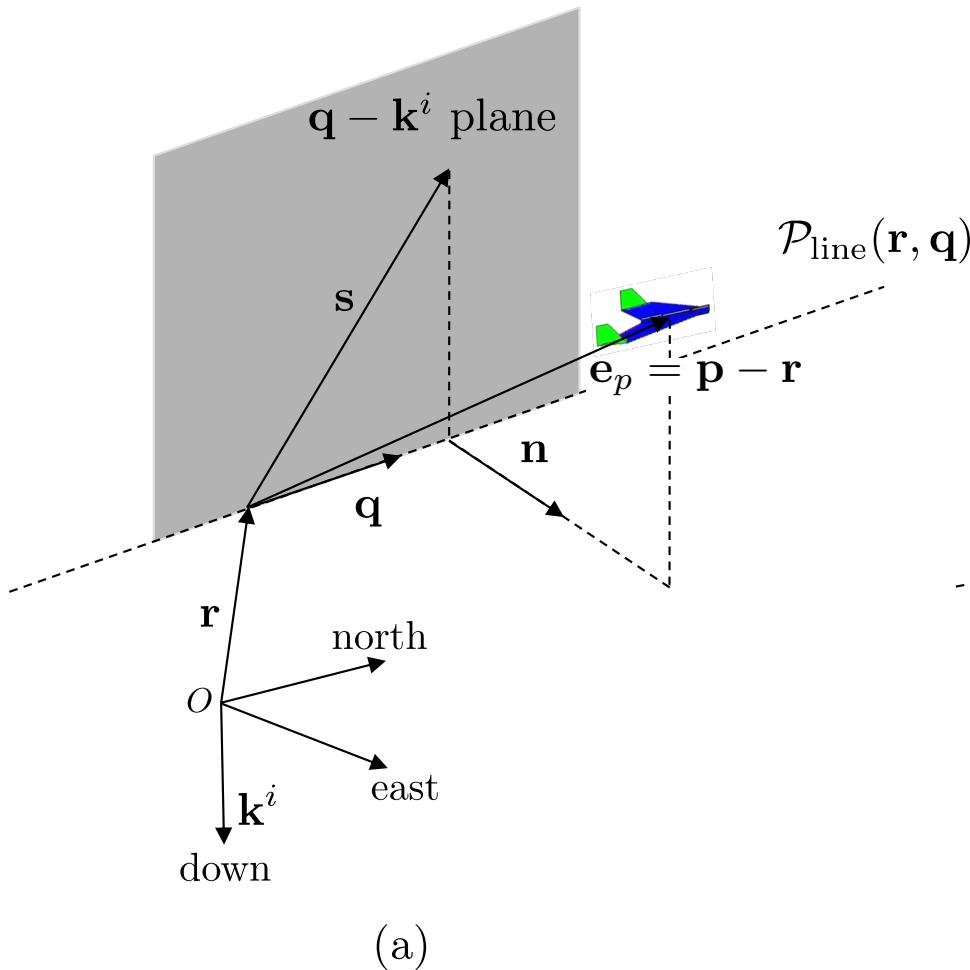
Assume that course controller works and $\chi = \chi_q + \chi^d(e_{py})$

Since

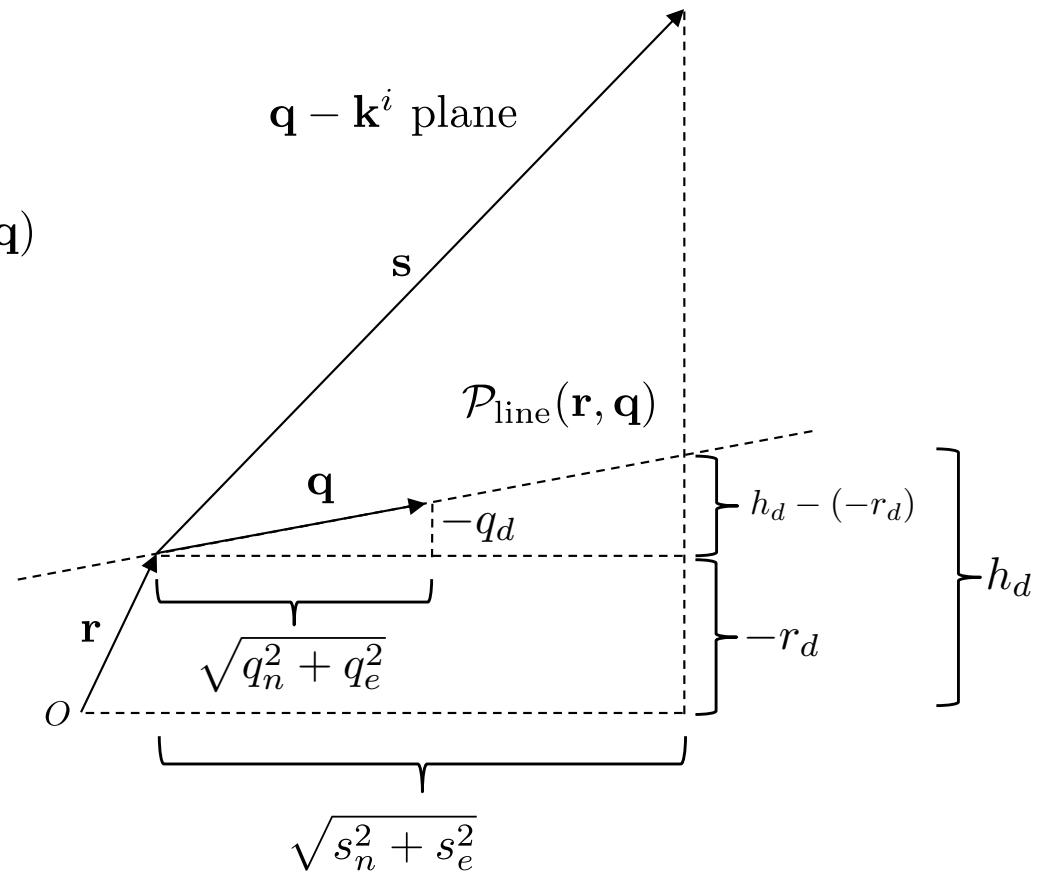
$$\begin{aligned}\dot{W} &= e_{py} \dot{e}_{py} \\ &= -V_a e_{py} \sin\left(\chi^\infty \frac{2}{\pi} \tan^{-1}(k_{\text{path}} e_{py})\right) \\ &< 0\end{aligned}$$

for $e_{py} \neq 0$, then $e_{py} \rightarrow 0$ asymptotically

Longitudinal Tracking Problem

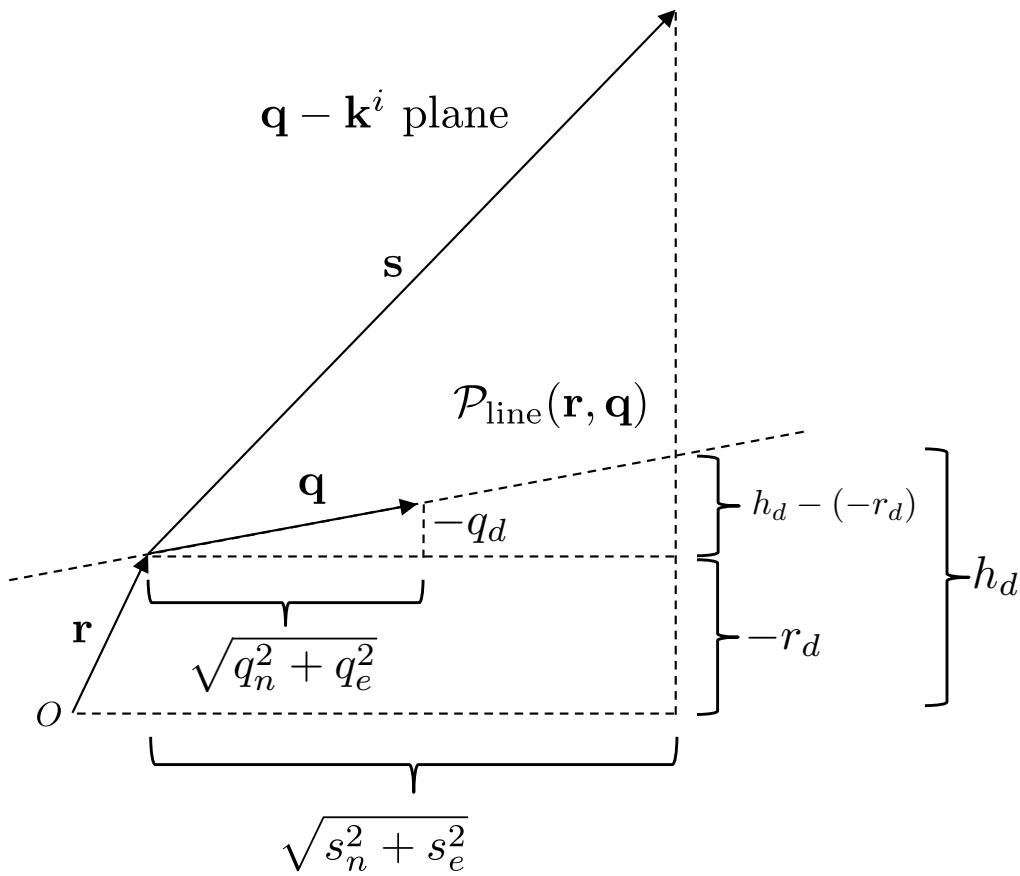


$$\mathbf{e}_p^i = \begin{pmatrix} e_{pn} \\ e_{pe} \\ e_{pd} \end{pmatrix} \triangleq \mathbf{p}^i - \mathbf{r}^i = \begin{pmatrix} p_n - r_n \\ p_e - r_e \\ p_d - r_d \end{pmatrix}$$



$$\begin{aligned} \mathbf{n} &= \frac{\mathbf{q} \times \mathbf{k}^i}{\|\mathbf{q} \times \mathbf{k}^i\|} & \mathbf{s}^i &= \begin{pmatrix} s_n \\ s_e \\ s_d \end{pmatrix} \\ &= \mathbf{e}_p^i - (\mathbf{e}_p^i \cdot \mathbf{n})\mathbf{n} \end{aligned}$$

Longitudinal Tracking Problem



(b)

By similar triangles

$$\frac{(h_d + r_d)}{\sqrt{s_n^2 + s_e^2}} = \frac{-q_d}{\sqrt{q_n^2 + q_e^2}}$$

Desired altitude based on current location

$$h_d(\mathbf{r}, \mathbf{p}, \mathbf{q}) = -r_d - \sqrt{s_n^2 + s_e^2} \left(\frac{q_d}{\sqrt{q_n^2 + q_e^2}} \right)$$

Select h^c so that $h \rightarrow h_d(\mathbf{r}, \mathbf{p}, \mathbf{q})$

Longitudinal Guidance Strategy

Use altitude state machine from Ch. 6.

Closed-loop altitude dynamics:

$$h(s) = \left(\frac{b_{\dot{h}}s + b_h}{s^2 + b_{\dot{h}}s + b_h} \right) h^c(s).$$

Altitude error:

$$e_h \doteq h_d(\mathbf{r}, \mathbf{p}, \mathbf{q}) - h = h^c - h$$

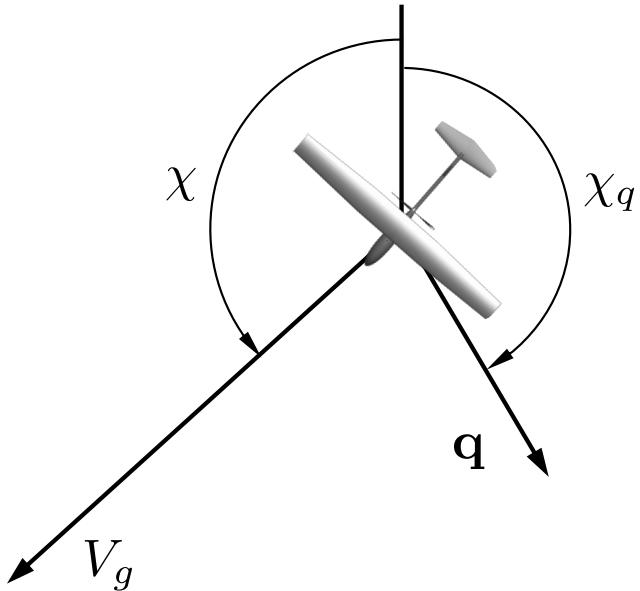
Error dynamics:

$$\begin{aligned} e_h(s) &= (1 - h(s)) h^c(s) \\ &= \left(\frac{s^2 + b_{\dot{h}}s + b_h}{s^2 + b_{\dot{h}}s + b_h} - \frac{b_{\dot{h}}s + b_h}{s^2 + b_{\dot{h}}s + b_h} \right) h^c(s) \\ &= \left(\frac{s^2}{s^2 + b_{\dot{h}}s + b_h} \right) h^c(s) \end{aligned}$$

Applying FVT:

$$\begin{aligned} e_{h,ss} &= \lim_{s \rightarrow 0} s \frac{s^2}{s^2 + b_{\dot{h}}s + b_h} h^c \\ &= 0, \quad \text{for } h^c = \frac{H_0}{s}, \frac{H_0}{s^2} \end{aligned}$$

Smallest Angle Turn Logic



$$\chi_q = \text{atan2}(q_e, q_n) + 2\pi m$$

$m \in \mathcal{N}$ is selected so that $-\pi \leq \chi_q - \chi \leq \pi$

Algorithm 3 Straight-line Following: $[h^c, \chi^c] = \text{followStraightLine}(\mathbf{r}, \mathbf{q}, \mathbf{p}, \chi)$

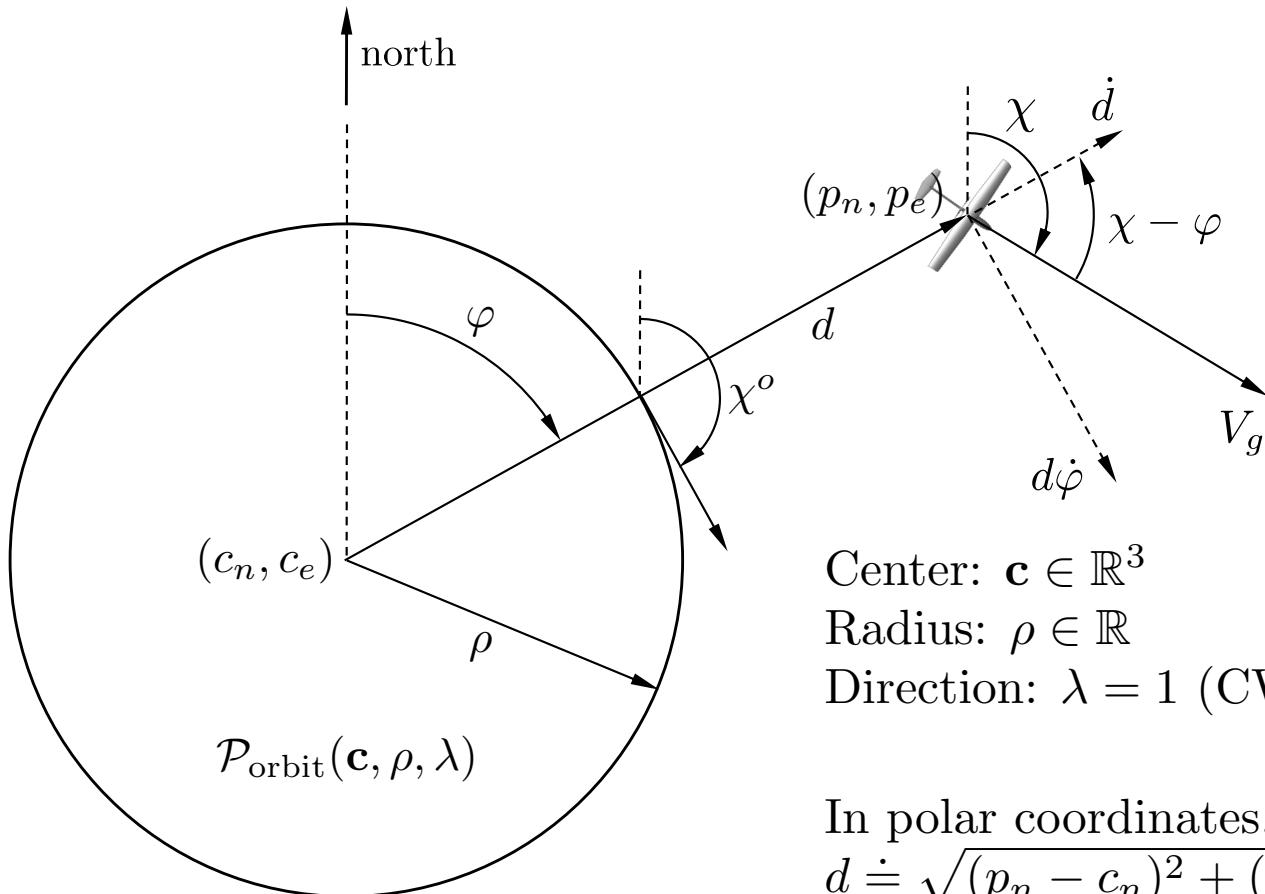
Input: Path definition $\mathbf{r} = (r_n, r_e, r_d)^\top$ and $\mathbf{q} = (q_n, q_e, q_d)^\top$, MAV position $\mathbf{p} = (p_n, p_e, p_d)^\top$, course χ , gains $\chi_\infty, k_{\text{path}}$, sample rate T_s .

- 1: Compute commanded altitude using equation (10.5).
- 2: $\chi_q \leftarrow \text{atan2}(q_e, q_n)$
- 3: **while** $\chi_q - \chi < -\pi$ **do**
- 4: $\chi_q \leftarrow \chi_q + 2\pi$
- 5: **end while**
- 6: **while** $\chi_q - \chi > \pi$ **do**
- 7: $\chi_q \leftarrow \chi_q - 2\pi$
- 8: **end while**
- 9: $e_{py} \leftarrow -\sin \chi_q (p_n - r_n) + \cos \chi_q (p_e - r_e)$
- 10: Compute commanded course angle using equation (10.8).
- 11: **return** h^c, χ^c

Orbit Following

Orbit definition:

$$\mathcal{P}_{\text{orbit}}(\mathbf{c}, \rho, \lambda) = \left\{ \mathbf{r} \in \mathbb{R}^3 : \mathbf{r} = \mathbf{c} + \lambda \rho \begin{pmatrix} \cos \varphi & \sin \varphi & 0 \end{pmatrix}^\top, \varphi \in [0, 2\pi) \right\}$$



Center: $\mathbf{c} \in \mathbb{R}^3$
 Radius: $\rho \in \mathbb{R}$
 Direction: $\lambda = 1$ (CW) or $\lambda = -1$ (CCW)

In polar coordinates, the position of MAV given by:
 $d \doteq \sqrt{(p_n - c_n)^2 + (p_e - c_e)^2}$:
 $\varphi \doteq \tan^{-1} \left(\frac{p_e - c_e}{p_n - c_n} \right)$:

Orbit Following

Easiest to analyze in polar coordinates.

Using

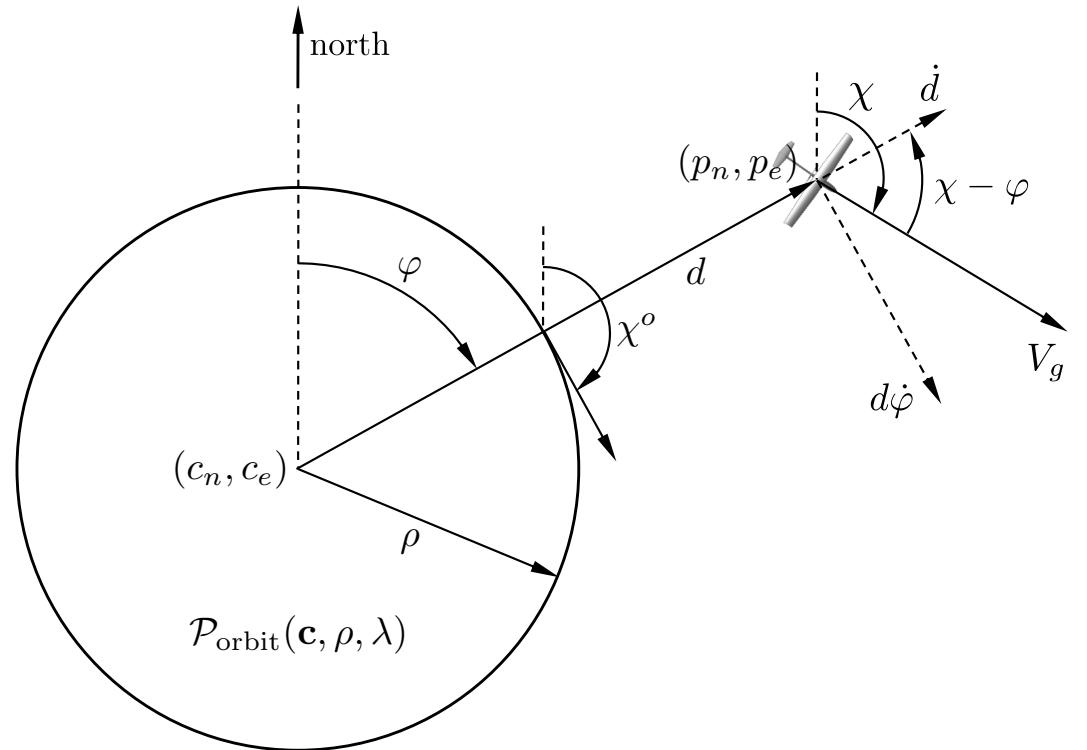
$$\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \end{pmatrix} = \begin{pmatrix} V_g \cos \chi \\ V_g \sin \chi \end{pmatrix}$$

and converting to polar coordinates gives

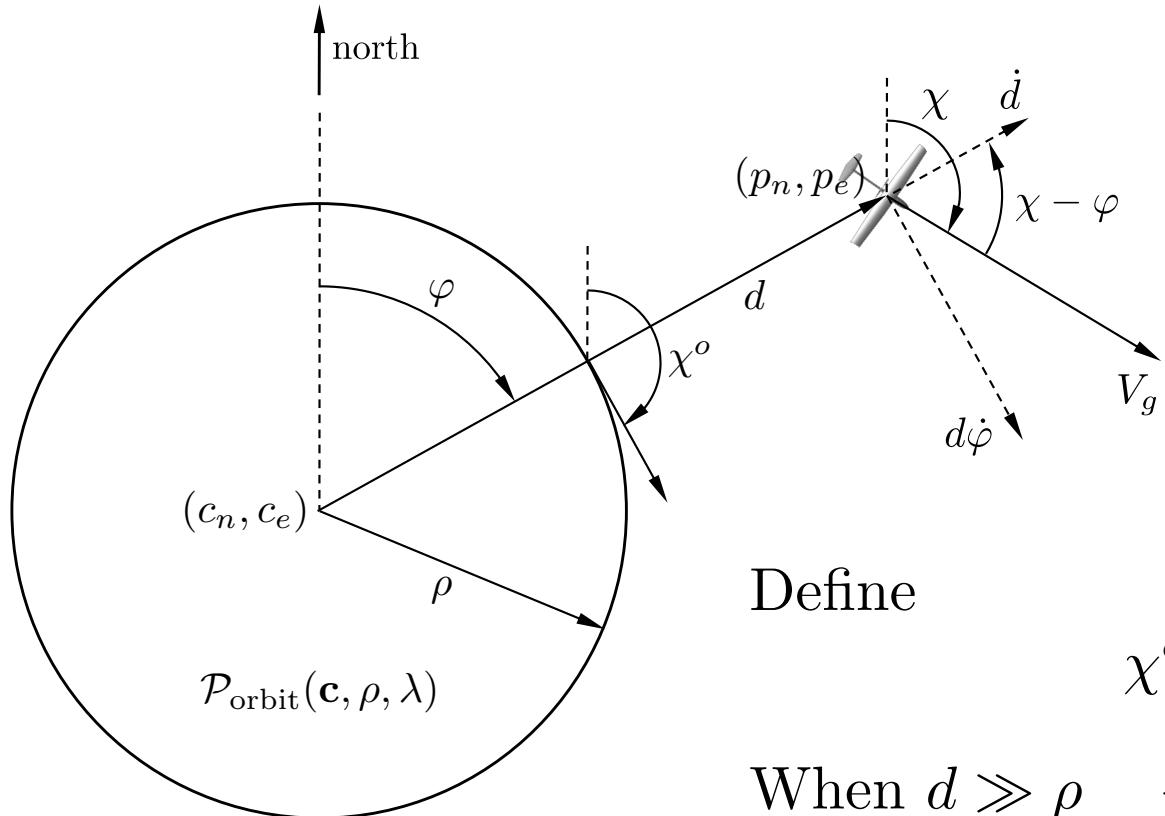
$$\begin{aligned} \begin{pmatrix} \dot{d} \\ d\dot{\varphi} \end{pmatrix} &= \begin{pmatrix} \cos \varphi & \sin \varphi \\ -\sin \varphi & \cos \varphi \end{pmatrix} \begin{pmatrix} \dot{p}_n \\ \dot{p}_e \end{pmatrix} \\ &= \begin{pmatrix} \cos \varphi & \sin \varphi \\ -\sin \varphi & \cos \varphi \end{pmatrix} \begin{pmatrix} V_g \cos \chi \\ V_g \sin \chi \end{pmatrix} \\ &= \begin{pmatrix} V_g \cos(\chi - \varphi) \\ V_g \sin(\chi - \varphi) \end{pmatrix} \end{aligned}$$

The result is

$$\begin{aligned} \dot{d} &= V_g \cos(\chi - \varphi) \\ \dot{\varphi} &= \frac{V_g}{d} \sin(\chi - \varphi) \\ \ddot{\chi} &= -b_{\dot{\chi}} \dot{\chi} + b_{\chi} (\chi^c - \chi) \end{aligned}$$



Orbit Following



Define

$$\chi^o = \varphi + \lambda \frac{\pi}{2}$$

When $d \gg \rho \rightarrow \chi_d \approx \chi^o + \lambda \frac{\pi}{2}$.

When $d = \rho \rightarrow \chi_d = \chi^o$

Therefore, let the desired course angle be

$$\chi_d(d-\rho, \lambda) = \chi^o + \lambda \tan^{-1} \left(k_{\text{orbit}} \left(\frac{d-\rho}{\rho} \right) \right)$$

Orbit Tracking Stability Analysis

Define the Lyapunov function $W = \frac{1}{2}(d - \rho)^2$

Assume that course controller works and $\chi = \chi^d(d - \rho, \lambda)$

Since

$$\begin{aligned}\dot{W} &= (d - \rho)\dot{d} \\ &= (d - \rho)(V_g \cos(\chi - \varphi)) \\ &= -V_g(d - \rho) \sin\left(\tan^{-1}\left(k_{\text{orbit}}\left(\frac{d - \rho}{\rho}\right)\right)\right) \\ &< 0\end{aligned}$$

for $d - \rho \neq 0$, then $d - \rho \rightarrow 0$ asymptotically

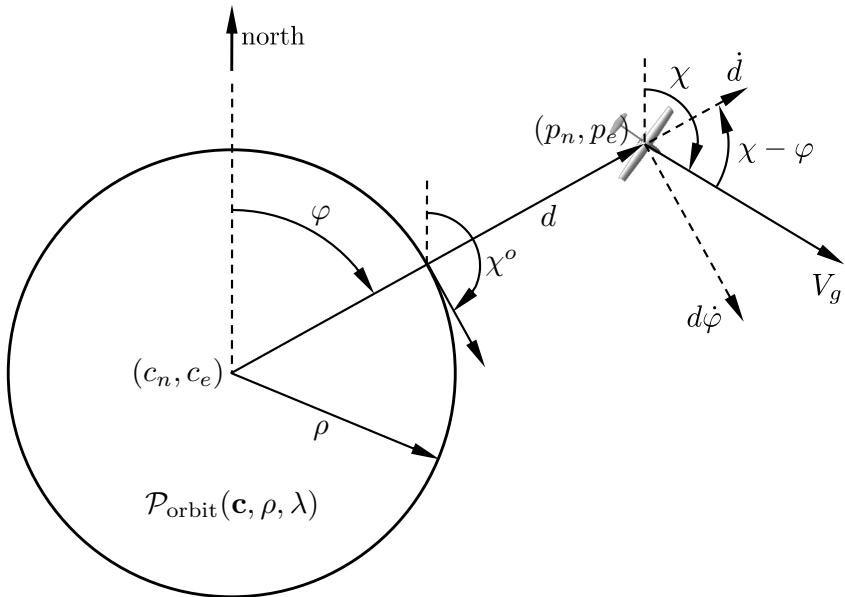
Orbit Following

The commanded course is

$$\chi^c(t) = \varphi + \lambda \left[\frac{\pi}{2} + \tan^{-1} \left(k_{\text{orbit}} \left(\frac{d - \rho}{\rho} \right) \right) \right]$$

The orbit angle must be wrapped:

$$\varphi = \text{atan2}(p_e - c_e, p_n - c_n) + 2\pi m$$



Algorithm 4 Circular Orbit Following: $[h^c, \chi^c] = \text{followOrbit}(\mathbf{c}, \rho, \lambda, \mathbf{p}, \chi)$

Input: Orbit center $\mathbf{c} = (c_n, c_e, c_d)^\top$, radius ρ , and direction λ , MAV position $\mathbf{p} = (p_n, p_e, p_d)^\top$, course χ , gains k_{orbit} , sample rate T_s .

- 1: $h^c \leftarrow -c_d$
- 2: $d \leftarrow \sqrt{(p_n - c_n)^2 + (p_e - c_e)^2}$
- 3: $\varphi \leftarrow \text{atan2}(p_e - c_e, p_n - c_n)$
- 4: **while** $\varphi - \chi < -\pi$ **do**
- 5: $\varphi \leftarrow \varphi + 2\pi$
- 6: **end while**
- 7: **while** $\varphi - \chi > \pi$ **do**
- 8: $\varphi \leftarrow \varphi - 2\pi$
- 9: **end while**
- 10: Compute commanded course angle using equation (10.13).
- 11: **return** h^c, χ^c

Roll Feedforward: no wind

For orbit following:

$$\chi^c(t) = \varphi + \lambda \left[\frac{\pi}{2} + \tan^{-1} \left(k_{\text{orbit}} \left(\frac{d - \rho}{\rho} \right) \right) \right].$$

Note:

$$\begin{aligned} d - \rho = 0 &\implies \chi^c = 0 \\ &\implies \phi^c = 0 \\ &\implies \text{UAV will immediately deviate from orbit.} \end{aligned}$$

Problem can be fixed by commanded a roll feedforward for when aircraft is on the orbit.

If on the orbit and no wind, then

$$\dot{\psi}^d = \lambda \frac{V_a}{\rho}.$$

Coordinated turn condition:

$$\dot{\psi} = \frac{g}{V_a} \tan \phi.$$

Equating and solving for roll gives

$$\phi_{ff} = \lambda \tan^{-1} \left(\frac{V_a^2}{g\rho} \right). \quad (1)$$

Roll Feedforward: wind

When wind is present we have

$$\dot{\chi}^d(t) = \lambda \frac{V_g(t)}{\rho},$$

where V_g is the time varying ground speed. The coordinated turn condition in wind is

$$\dot{\chi} = \frac{g}{V_g} \tan \phi \cos(\chi - \psi).$$

Equating these expressions and solving for ϕ gives

$$\phi_{ff} = \lambda \tan^{-1} \left(\frac{V_g^2}{g\rho \cos(\chi - \psi)} \right).$$

Or equivalently

$$\phi_{ff} = \lambda \tan^{-1} \left(\frac{\left((w_n \cos \chi + w_e \sin \chi) + \sqrt{V_a^2 - (w_n \sin \chi - w_e \cos \chi)^2 - w_d^2} \right)^2}{g\rho \sqrt{\frac{V_a^2 - (w_n \sin \chi - w_e \cos \chi)^2 - w_d^2}{V_a^2 - w_d^2}}} \right),$$

Dubins Airplane Model

Adapted from: Mark Owen, Randal W. Beard, Timothy W. McLain, "Implementing Dubins Airplane Paths on Fixed-wing UAVs," *Handbook of Unmanned Aerial Vehicles*, ed. Kimon P. Valavanis, George J. Vachtsevanos, Springer Verlag, Section XII, Chapter 68, p. 1677-1702, 2014.

Dubins Airplane model:

$$\dot{r}_n = V \cos \psi \cos \gamma^c$$

$$\dot{r}_e = V \sin \psi \cos \gamma^c$$

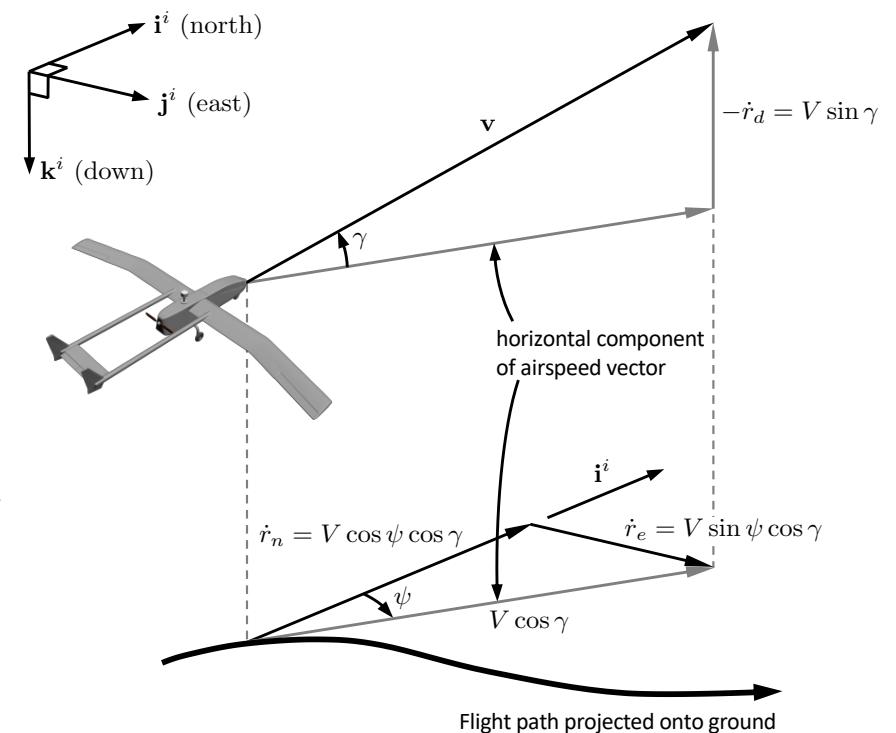
$$\dot{r}_d = -V \sin \gamma^c$$

$$\dot{\psi} = \frac{g}{V} \tan \phi^c$$

Where the commanded flight path angle γ^c and the commanded roll angle ϕ^c are constrained by

$$|\phi^c| \leq \bar{\phi}$$

$$|\gamma^c| \leq \bar{\gamma}.$$



3D Vector Field Path Following

Adapted from: V. M. Goncalves, L. C. A. Pimenta, C. A. Maia, B. C. O. Durtra, G. A. S. Pereira, B. C. O. Dutra, and G. A. S. Pereira, "Vector Fields for Robot Navigation Along Time-Varying Curves in n-Dimensions," IEEE Transactions on Robotics, vol. 26, pp. 647–659, Aug 2010.

The path is specified as the intersection of two 2D manifolds given by

$$\begin{aligned}\alpha_1(\mathbf{r}) &= 0 \\ \alpha_2(\mathbf{r}) &= 0\end{aligned}$$

$\mathbf{r} \in \mathbb{R}^3$. Define the composite function

$$W(\mathbf{r}) = \frac{1}{2}\alpha_1^2(\mathbf{r}) + \frac{1}{2}\alpha_2^2(\mathbf{r}),$$

Note that the gradient

$$\frac{\partial W}{\partial \mathbf{r}} = \alpha_1(\mathbf{r}) \frac{\partial \alpha_1}{\partial \mathbf{r}}(\mathbf{r}) + \alpha_2(\mathbf{r}) \frac{\partial \alpha_2}{\partial \mathbf{r}}(\mathbf{r}).$$

points away from the path.

3D Vector Field Path Following

The desired velocity vector can be chosen as

$$\mathbf{u}' = \underbrace{-K_1 \frac{\partial W}{\partial \mathbf{r}}}_{\text{velocity directed toward the path}} + \underbrace{K_2 \frac{\partial \alpha_1}{\partial \mathbf{r}} \times \frac{\partial \alpha_2}{\partial \mathbf{r}}}_{\text{velocity directed along the path}}$$

where $K_1 > 0$ and K_2 are symmetric tuning matrices, and the definiteness of K_2 determines the direction of travel along the path.

Since \mathbf{u}' may not equal V_a , normalize to get

$$\mathbf{u} = V_a \frac{\mathbf{u}'}{\|\mathbf{u}'\|}.$$

3D Vector Field Path Following

Setting the NED components of the velocity of the Dubins airplane model to $\mathbf{u} = (u_1, u_2, u_3)^\top$ gives

$$\begin{aligned} V \cos \psi^d \cos \gamma^c &= u_1 \\ V \sin \psi^d \cos \gamma^c &= u_2 \\ -V \sin \gamma^c &= u_3. \end{aligned}$$

Solving for γ^c , and ψ^d results in

$$\begin{aligned} \gamma^c &= -\text{sat}_{\bar{\gamma}} \left[\sin^{-1} \left(\frac{u_3}{V} \right) \right] \\ \psi^d &= \text{atan2}(u_2, u_1). \end{aligned}$$

Assuming the inner-loop lateral-directional dynamics are accurately modeled by the coordinated-turn equation, the commanded roll angle is

$$\phi^c = \text{sat}_{\bar{\phi}} [k_\phi (\psi^d - \psi)],$$

where k_ϕ is a positive constant.

3D Vector Field – Straight Line

The straight line path is given by **path**

$$\mathcal{P}_{\text{line}}(\mathbf{c}_\ell, \psi_\ell, \gamma_\ell) = \left\{ \mathbf{r} \in \mathbb{R}^3 : \mathbf{r} = \mathbf{c}_\ell + \sigma \mathbf{q}_\ell, \sigma \in \mathbb{R} \right\},$$

where

$$\mathbf{q}_\ell = \begin{pmatrix} q_n \\ q_e \\ q_d \end{pmatrix} = \begin{pmatrix} \cos \psi_\ell \cos \gamma_\ell \\ \sin \psi_\ell \cos \gamma_\ell \\ -\sin \gamma_\ell \end{pmatrix}.$$

Define

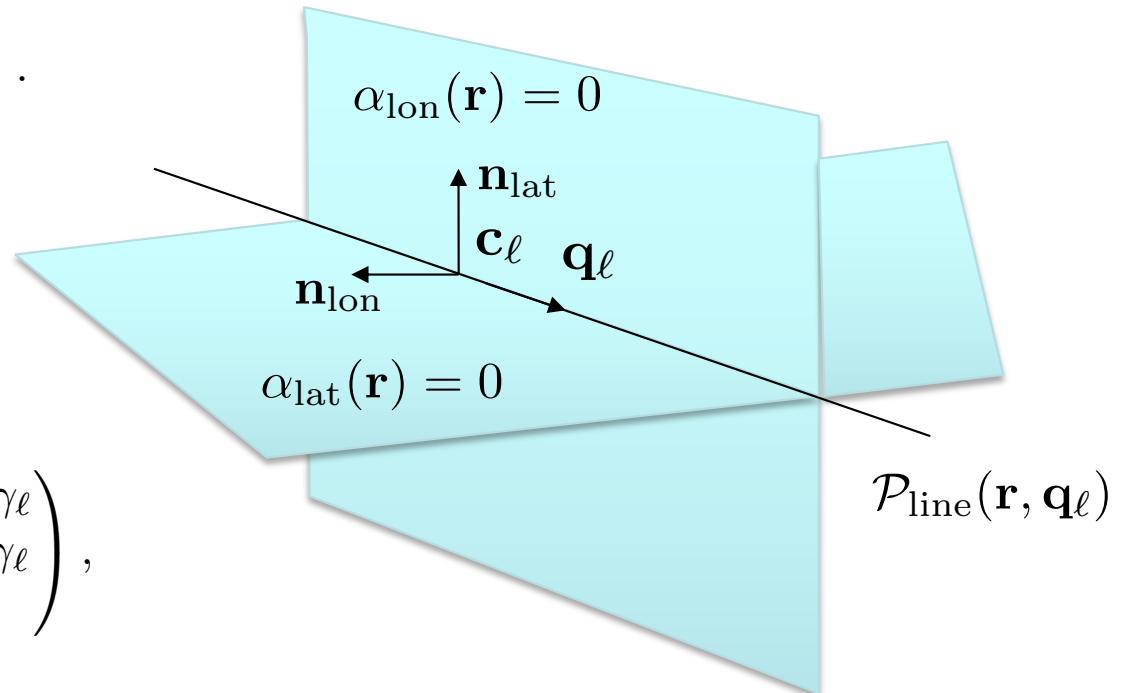
$$\mathbf{n}_{\text{lon}} = \begin{pmatrix} -\sin \psi_\ell \\ \cos \psi_\ell \\ 0 \end{pmatrix}$$

$$\mathbf{n}_{\text{lat}} = \mathbf{n}_{\text{lon}} \times \mathbf{q}_\ell = \begin{pmatrix} -\cos \psi_\ell \sin \gamma_\ell \\ -\sin \psi_\ell \sin \gamma_\ell \\ -\cos \gamma_\ell \end{pmatrix},$$

to get

$$\alpha_{\text{lon}}(\mathbf{r}) = \mathbf{n}_{\text{lon}}^\top (\mathbf{r} - \mathbf{c}_\ell) = 0$$

$$\alpha_{\text{lat}}(\mathbf{r}) = \mathbf{n}_{\text{lat}}^\top (\mathbf{r} - \mathbf{c}_\ell) = 0.$$



3D Vector Field – Helical Path

A helical path is then defined as

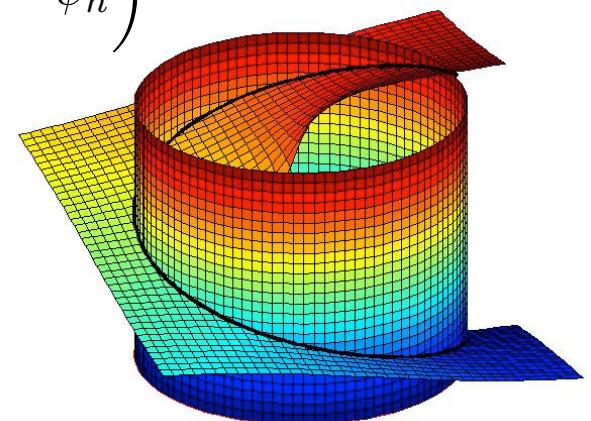
$$\mathcal{P}_{\text{helix}}(\mathbf{c}_h, \psi_h, \lambda_h, \rho_h, \gamma_h) = \{\mathbf{r} \in \mathbb{R}^3 : \alpha_{\text{cyl}}(\mathbf{r}) = 0 \text{ and } \alpha_{\text{pl}}(\mathbf{r}) = 0\}.$$

where

$$\begin{aligned}\alpha_{\text{cyl}}(\mathbf{r}) &= \left(\frac{r_n - c_n}{\rho_h} \right)^2 + \left(\frac{r_e - c_e}{\rho_h} \right)^2 - 1 \\ \alpha_{\text{pl}}(\mathbf{r}) &= \left(\frac{r_d - c_d}{\rho_h} \right) + \frac{\tan \gamma_h}{\lambda_h} \left(\tan^{-1} \left(\frac{r_e - c_e}{r_n - c_n} \right) - \psi_h \right)\end{aligned}$$

where the initial position along the helix is

$$\mathbf{r}(0) = \mathbf{c}_h + \begin{pmatrix} \rho_h \cos \psi_h \\ \rho_h \sin \psi_h \\ 0 \end{pmatrix},$$



and where \mathbf{c}_h is the center of the helix, ρ_h is the radius, γ_h is the climb angle.