Terminal Guidance Laws: A Comprehensive Survey from Classical Methods to Learning-Based Approaches

Rasit Evduzen
Independent Researcher
github.com/RasitEvduzen

August 2026

Abstract

This survey reviews terminal guidance laws for guided munitions within a single, unified framework. Starting from the classical proportional navigation family (True, Pure, and Augmented Proportional Navigation), the review proceeds through practical corrections such as gravity and drag compensation, the generalized vector explicit guidance law (GENEX) that additionally controls the terminal impact angle, sliding-mode-based robust guidance, differential game guidance grounded in the Hamilton-Jacobi-Isaacs equation, and finally reinforcement learning based guidance. For each method the underlying derivation logic, the command equations, and the design trade-offs are presented with numbered equations. The methods are compared along the axes of information requirement, robustness, computational load, and theoretical guarantee. A recurring theme is that classical laws and learning-based methods are connected through the common zero-effort-miss and dynamic-programming principles, which motivates the current trend toward hybrid designs that combine the analytic reliability of classical laws with the flexibility of learning-based methods.

Keywords: Terminal guidance, proportional navigation, optimal guidance, sliding mode control, differential games, reinforcement learning, impact angle control.

1 Introduction

The flight of a guided munition is commonly divided into three phases: launch, midcourse, and terminal homing. The terminal phase, from seeker acquisition to intercept, is the critical stage in which the miss distance is determined. During this phase the seeker provides measurements such as the line-of-sight (LOS) angle, range, and range rate. The task of the guidance law is to convert these measurements into an acceleration command that intercepts the target with the smallest possible miss distance, frequently while satisfying additional constraints such as impact angle, impact time, or field-of-view limits [3, 4].

Classical guidance laws have been used for decades due to their simplicity and reliability, yet they can be limited against highly maneuverable targets and uncertain conditions. This has motivated the development of new laws based on optimal control, robust control, game theory, and machine learning. The purpose of this survey is to present this spectrum within one framework, using a common notation and a comparative perspective. A broad taxonomy of recent guidance, navigation, and control research is given in [19].

The paper is organized as follows. Section 2 introduces the common concepts and the engagement geometry. Section 3 treats the classical proportional navigation family. Section 4 discusses practical corrections. Section 5 presents generalized explicit guidance, Section 6 sliding mode guidance, and Section 7 differential game guidance. Section 8 presents learning-based guidance. Section 9 provides a comparative discussion, and Section 10 concludes.

2 Preliminaries and Engagement Geometry

Most terminal guidance laws are built on a common set of quantities. The line-of-sight (LOS) is the line connecting the missile and the target; its angle relative to an inertial reference is denoted λ, and its time derivative, the LOS rate, is $\dot{\lambda}$. The range R is the missile-to-target distance, and the closing velocity is $V_c = -\dot{R}$.

The fundamental principle of terminal guidance is the collision-triangle concept: if the LOS angle remains constant, i.e. $\dot{\lambda} = 0$, the missile and target approach at a fixed bearing and collision becomes inevitable. This straight-line course is also the shortest path, hence the trajectory requiring the least time and the least control effort. Most classical guidance laws therefore aim to drive the LOS rate to zero.

To evaluate the methods within a common framework, the zero-effort-miss (ZEM) concept is useful. The ZEM is the miss distance that would result if no further maneuver were applied from the current instant. In terms of the time-to-go t_{go}, the LOS-normal relative position y, and its rate $\dot{y}$, the ZEM is

$$\mathrm{ZEM} = y + \dot{y}\, t_{go}. \tag{1}$$

The following sections show how various laws are derived through this common term.

3 The Proportional Navigation Family

Proportional navigation (PN) is the foundational law of terminal guidance. It issues a lateral acceleration command proportional to the LOS rate, thereby maintaining the collision course:

$$n_c = N'\, V_c\, \dot{\lambda}, \tag{2}$$

where n_c is the acceleration command, N' is the effective navigation ratio (typically $3 \leq N' \leq 5$), V_c is the closing velocity, and $\dot{\lambda}$ is the LOS rate. Using (1), PN can also be written in ZEM form:

$$n_c = \frac{N'}{t_{go}^2}\, \mathrm{ZEM}. \tag{3}$$

This form is the point of departure for later generalizations. The family members differ in the axis along which the command is applied and in the scaling velocity. The baseline PN guidance loop is summarized in Algorithm 1.

Algorithm 1 Proportional Navigation (PN).

1: Select the navigation ratio N' (typically $3 \leq N' \leq 5$)
2: **for** each guidance cycle **do**
3: Read seeker measurements: LOS angle λ, range R, range rate $\dot{R}$
4: Compute the LOS rate $\dot{\lambda}$ and the closing velocity $V_c = -\dot{R}$
5: Compute the command $n_c = N' V_c \dot{\lambda}$
6: Saturate to $a_{\max}$ and send the command to the autopilot
7: **end for**

3.1 True Proportional Navigation (TPN)

TPN applies the acceleration command normal to the LOS, scaled by the closing velocity:

$$a_{\text{TPN}} = N' V_c \dot{\lambda} \qquad \text{(normal to LOS)}. \tag{4}$$

Fixing the command direction to the LOS makes the equations linear, which allows closed-form analysis of miss distance and capturability. However, an LOS-normal command can have a component along the missile velocity vector; since an aerodynamic missile cannot produce such a component, TPN is primarily an analytic reference model. The TPN guidance loop is summarized in Algorithm 2.

Algorithm 2 True Proportional Navigation (TPN).

1: Select the navigation ratio N'
2: **for** each guidance cycle **do**
3: Measure the LOS rate $\dot{\lambda}$ and the closing velocity V_c
4: Compute the command magnitude $a_{\text{TPN}} = N' V_c \dot{\lambda}$
5: Direct the command normal to the line of sight (LOS)
6: Saturate to $a_{\max}$ and send the command to the autopilot
7: **end for**

3.2 Pure Proportional Navigation (PPN)

PPN applies the acceleration normal to the missile velocity vector, scaled by the missile speed:

$$a_{\text{PPN}} = N' V_m \dot{\lambda} \qquad \text{(normal to velocity)}. \tag{5}$$

Since aerodynamic force (lift) is always produced normal to the velocity vector, PPN is directly realizable and is the form implemented in most practical missiles. Its geometric essence is that it rotates the velocity vector at a rate proportional to the LOS rate:

$$\dot{\gamma} = N' \dot{\lambda}, \tag{6}$$

where γ is the missile flight-path angle. Equation (6) states that for each unit rotation of the LOS the missile rotates its velocity vector by a factor N'. The PPN guidance loop is summarized in Algorithm 3.

Algorithm 3 Pure Proportional Navigation (PPN).

1: Select the navigation ratio N'
2: **for** each guidance cycle **do**
3: Measure the LOS rate $\dot{\lambda}$ and the missile speed V_m
4: Compute the command magnitude $a_{\mathrm{PPN}} = N' V_m \dot{\lambda}$
5: Direct the command normal to the missile velocity vector
6: Saturate to $a_{\max}$ and send the command to the autopilot
7: **end for**

3.3 Augmented Proportional Navigation (APN)

Classical PN yields zero miss against a non-maneuvering target, but if the target maneuvers with acceleration the required command grows toward the end of flight. APN adds the target acceleration as a feed-forward term, removing this lag [3]:

$$a_{\mathrm{APN}} = N' V_c \dot{\lambda} + \frac{N'}{2} a_T, \tag{7}$$

where a_T is the LOS-normal component of the target acceleration. APN is also obtained as the solution of a linear-quadratic optimal control problem in which the target acceleration is assumed constant; in this case the ZEM term is extended:

$$\mathrm{ZEM}_{\mathrm{APN}} = y + \dot{y}\, t_{go} + \tfrac{1}{2}\, a_T\, t_{go}^2. \tag{8}$$

Comparing (7) with (8), setting $a_T = 0$ reduces APN to classical PN, i.e. PN is a special case of APN. APN substantially reduces the peak acceleration demand and miss distance against maneuvering targets, at the cost of requiring an estimate of a_T; the estimation error directly affects performance. The APN guidance loop is summarized in Algorithm 4.

Algorithm 4 Augmented Proportional Navigation (APN).

1: Select the navigation ratio N'; initialize the target-acceleration estimator
2: **for** each guidance cycle **do**
3: Measure the LOS rate $\dot{\lambda}$ and the closing velocity V_c
4: Estimate the target acceleration a_T normal to the LOS
5: Compute the command $a_{\mathrm{APN}} = N' V_c \dot{\lambda} + \frac{N'}{2} a_T$
6: (Optional) add the gravity bias $-g_\perp$ with $g_\perp \approx g \cos\gamma$
7: Saturate to $a_{\max}$ and send the command to the autopilot
8: **end for**

4 Practical Corrections

The derivations above rely on idealized assumptions such as a flat earth, point masses, constant speed, and negligible gravity and drag. In real flight two known forces act on the missile and can be compensated by feed-forward.

4.1 Gravity Compensation

If gravity is not compensated, the missile falls below the collision course and a persistent miss component results. Because gravity is known and predictable, a bias that cancels its component normal to the relevant plane is added to the command:

$$a_{\text{cmd}} = N' V_c \dot{\lambda} - g_\perp, \qquad g_\perp \approx g \cos\gamma. \tag{9}$$

The PN term then operates as if in a gravity-free environment. If both missile and target are in free fall, the net contribution to the relative geometry is reduced; for a powered missile that must hold altitude, compensation is required.

4.2 Drag and Variable Speed

Drag continuously reduces the speed and hence the closing velocity, which lengthens t_{go} and introduces error in formulas that assume constant speed. In practice V_c is measured or estimated online, and the predictable component of drag can also be accounted for by feed-forward.

4.3 Time-to-Go Estimation

ZEM-based laws depend on t_{go}; because the denominator in (3) and (8) is t_{go}^2, the error grows near intercept. The simplest estimate is the range divided by the closing velocity:

$$t_{go} \approx \frac{R}{V_c}. \tag{10}$$

Corrections that account for the heading error and optimal-control-based estimators have been proposed for curved trajectories. In addition, the finite response time of the autopilot (time constant τ) increases the miss distance, and N' is selected with this lag in mind.

5 Generalized Explicit Guidance (GENEX)

GENEX (generalized vector explicit guidance) is an explicit guidance law that, in addition to nulling the miss distance, controls the direction of the terminal velocity vector, i.e. the impact angle [1]. The impact angle is important for warhead effectiveness; for example, fragmentation warheads and topside attacks against surface targets require a specific terminal geometry.

5.1 Cost Function and Generalization

GENEX generalizes the standard control-energy cost through a design parameter:

$$J = \int_0^{T_0} \frac{u^2}{2\,T^n}\,dT, \qquad T = t_f - t, \quad n \in \mathbb{Z}^{\geq 0}. \tag{11}$$

The factor T^n in the denominator penalizes control usage increasingly as the time-to-go decreases; the effect strengthens as n grows. GENEX therefore defines not a single cost but a family of costs parameterized by n.

5.2 Derivation and General Solution

For the linear system $\dot{X} = AX + bu$ with terminal condition $X(t_f) = \mathbf{0}$, Pontryagin's minimum principle, together with the fundamental matrix M and $Q(T) = \int_0^T (Mb)(Mb)^\top T^n\,dT$, yields the general optimal control

$$u^* = -(Mb)^\top Q^{-1} M X\, T^n. \tag{12}$$

When the state is taken as the ZEM alone, the result recovers the classical PN command:

$$u^* = \frac{n+3}{T^2}\,\mathrm{ZEM}, \tag{13}$$

so the navigation ratio is $\Lambda = n + 3$; for $n = 0$ this gives $\Lambda = 3$, i.e. standard PN. Thus PN is the simplest member of the GENEX family.

5.3 Full Law and Physical Interpretation

When a terminal velocity constraint is added to the miss-distance objective, the optimal control becomes a two-term law using two states (the ZEM relative to the specified final position and the difference from the desired final velocity):

$$u^* = \frac{1}{T^2}\Big[K_1\,(y_f - y_M - \dot{y}_M T) + K_2\,(\dot{y}_f - \dot{y}_M)\,T\Big], \tag{14}$$

$$K_1 = (n+2)(n+3), \qquad K_2 = -(n+1)(n+2). \tag{15}$$

The first term (through K_1) behaves like PN and drives the miss distance to zero; the second term (through K_2) is proportional to the difference between the current and desired terminal flight-path angles and enforces the impact angle. The vector and single-plane forms, with range R, average remaining speed V, and heading error δ, are

$$u = \frac{V^2}{R}\Big[K_1(\hat{r} - \hat{v}) + K_2(\hat{v}_f - \hat{v})\Big], \tag{16}$$

$$u = \frac{V^2}{R}\Big[-K_1 \sin\delta - K_2 \sin(\gamma - \gamma_f)\Big]. \tag{17}$$

The design parameter n controls the curvature of the trajectory. Small n yields an early, low-acceleration, more direct path that may fail to achieve the desired impact angle; large n

achieves the angle but demands higher acceleration and risks saturation. Apart from using terminal-angle information instead of a_T, GENEX shares the same feed-forward philosophy as APN; both APN and PN are special cases of GENEX. The guidance loop is summarized in Algorithm 5.

Algorithm 5 GENEX Guidance.

1: Select the design parameter n and the desired terminal flight-path angle γ_f
2: **for** each guidance cycle **do**
3: Read states; estimate the time-to-go $T = t_{go} \approx R/V_c$
4: Compute the ZEM to the specified final position and the terminal velocity error
5: Set the gains $K_1 = (n+2)(n+3)$ and $K_2 = -(n+1)(n+2)$
6: Compute the command $u^* = \frac{1}{T^2}[K_1(y_f - y_M - \dot{y}_M T) + K_2(\dot{y}_f - \dot{y}_M)T]$
7: Saturate to $a_{\max}$ and send the command to the autopilot
8: Update the predicted intercept point, ZEM, and t_{go} for the next cycle
9: **end for**

6 Sliding Mode Guidance

Sliding mode control (SMC) is a variable structure control method [5, 2]. The system states are first driven onto a sliding surface and then kept on it; while on the surface the dynamics depend only on the surface definition, and model uncertainties and external disturbances are largely suppressed. This property gives SMC strong robustness.

6.1 Sliding Surface and Reaching Condition

A general sliding surface, for system order n and positive coefficient c, is

$$s(x,t) = \left(\frac{d}{dt} + c\right)^{n-1} x. \tag{18}$$

Reaching and remaining on the surface is guaranteed through a Lyapunov function:

$$V(s) > 0, \qquad \dot{V}(s) < 0. \tag{19}$$

6.2 Application to Guidance

In the guidance problem the sliding surface is chosen directly as the LOS rates:

$$s_1 = \dot{\theta}, \qquad s_2 = \dot{\phi}. \tag{20}$$

Driving this surface to zero means $\dot{\theta} = \dot{\phi} = 0$, i.e. locking onto the collision course. This is a robust realization of the basic aim of proportional navigation. In the LOS accelerations derived from the engagement kinematics the target accelerations are unknown; if the matching condition holds, these unknowns are lumped into a single uncertainty term:

$$\ddot{\theta} = \frac{-2\dot{r}\dot{\theta} - r\dot{\phi}^2 \cos\theta \sin\theta + D_1 - a_{m\theta}}{r}, \qquad \|D_1\| \le c_{10}, \tag{21}$$

where D_1 is the lumped uncertainty. Thus the exact target maneuver need not be known; only an upper bound on it is required. This is an important practical advantage over APN, which requires estimating the target acceleration.

6.3 Control Law, Chattering, and Adaptation

The acceleration command is chosen to include a signum function:

$$n_\theta = \frac{-2\dot{r}\dot{\theta} - r\dot{\phi}^2 \cos\theta \sin\theta + k_1 \ \mathrm{sgn}(s_1)}{\cos(\gamma_{m/\theta} - \theta)}. \tag{22}$$

Substituting this command yields the closed-loop dynamics

$$\dot{s}_1 = \frac{D_1 - k_1 \ \mathrm{sgn}(s_1)}{r}. \tag{23}$$

From (23), when $k_1 > D_1$ holds, $s_1 \dot{s}_1 < 0$ is achieved and the reaching condition is satisfied, i.e. the LOS rate is driven to zero. Because the signum function cannot switch ideally, the states oscillate at high frequency around the surface; this undesirable effect is called chattering. Chattering is mitigated by a boundary layer (replacing the signum with a smooth saturation), by super-twisting and higher-order sliding modes [11, 12], or by terminal SMC that provides finite-time convergence [9, 10]. If the uncertainty bound is unknown, the gain is tuned online through an adaptation law:

$$k_1 = \bar{c}_{10} + c_{11}|s_1|, \tag{24}$$

and stability is proven with the Lyapunov candidate

$$V = \tfrac{1}{2}\big(r^2(s_1^2 + s_2^2) + q_{10}^{-1}\tilde{c}_{10}^2 + q_{20}^{-1}\tilde{c}_{20}^2\big). \tag{25}$$

Whereas optimal laws such as GENEX rely on target and model information, SMC offers a different philosophy that is largely indifferent to uncertainty; the two approaches can also be combined in a hybrid design. The guidance loop is summarized in Algorithm 6.

Algorithm 6 Sliding Mode Guidance (with adaptation).

1: Choose the sliding surfaces $s_1 = \dot{\theta}$, $s_2 = \dot{\phi}$; initialize the adaptive gains
2: **for** each guidance cycle **do**
3: Read states; compute the LOS rates $s_1 = \dot{\theta}$ and $s_2 = \dot{\phi}$
4: Update the gains by the adaptation law $k_1 = \bar{c}_{10} + c_{11}|s_1|$ (and likewise k_2)
5: Compute the command from the control law, replacing $\mathrm{sgn}(\cdot)$ by a saturation $\mathrm{sat}(\cdot/\varepsilon)$ to limit chattering
6: Saturate to $a_{\max}$ and send the command to the autopilot
7: **end for**

7 Differential Game Guidance

The previous laws treat the target as passive or as a bounded disturbance. Differential game guidance (DGL) instead models the target as an intelligent adversary that optimizes its own evasion against the missile's moves [15, 14]. This is a pursuit-evasion problem, and because the two players have exactly opposing objectives it is a zero-sum game.

7.1 Min-Max Formulation

For a two-input system $\dot{x} = f(t, x, u, v)$ (with u the missile control and v the target control), the performance functional and solution are

$$J = q(x(t_f)) + \int_0^{t_f} g(t, x, u, v)\, dt, \qquad \min_u \max_v J. \tag{26}$$

The missile minimizes the cost (usually the capture time or terminal miss) while the target maximizes it. The solution sought is a saddle point at which no unilateral deviation benefits either player.

7.2 The Hamilton-Jacobi-Isaacs Equation

The theoretical basis of DGL is the Hamilton-Jacobi-Isaacs (HJI) equation, the extension of the single-decision-maker Hamilton-Jacobi-Bellman equation to two adversarial decision makers. The value function $V(t, x)$ represents the outcome under optimal play by both sides (for example the capture time) and satisfies the partial differential equation

$$-\frac{\partial V}{\partial t} = \min_u \max_v \left[\frac{\partial V}{\partial x} f(x, u, v) + g(x, u, v) \right]. \tag{27}$$

The bracketed expression is the Hamiltonian; the only difference from the HJB equation is the additional $\max_v$ next to $\min_u$. The gradient of the value function $\partial V/\partial x$ plays the role of the costate in Pontryagin's principle, so the Pontryagin derivation of GENEX and the HJI equation are two expressions of the same problem. The equality $\min_u \max_v = \max_v \min_u$ (the Isaacs condition) guarantees the existence of a saddle point. Because HJI is a nonlinear equation in as many variables as the state dimension, an analytic solution is possible only in special cases; this is the main source of the high computational load of DGL.

7.3 Missile-Specific Formulation

With linearized kinematics and the ZEM as the state, the bounded accelerations of both sides are taken into account:

$$|a_M| \le a_M^{\max}, \qquad |a_T| \le a_T^{\max}. \tag{28}$$

With the terminal miss as the payoff, the optimal strategies are bang-bang and depend on the sign of the ZEM:

$$a_M^* = a_M^{\max}\ \mathrm{sgn}(\mathrm{ZEM}), \qquad a_T^* = a_T^{\max}\ \mathrm{sgn}(\mathrm{ZEM}). \tag{29}$$

An ideal missile (DGL/0) and a first-order-lag missile (DGL/1) are defined; DGL/1 introduces a switching structure and a critical time-to-go threshold. The result is a capture zone within which a guaranteed hit is possible. In the simple-motion case the capture regions are described by Apollonius circles, with speed ratio $\beta = v_P/v_E > 1$:

$$(x - a_i)^2 + (y - b_i)^2 = \rho_i^2. \tag{30}$$

The DGL framework extends naturally to cooperative multi-missile capture (with Voronoi-based task allocation) and to the attacker-target-defender triad, which underlies active protection systems [18]. The guidance loop for the first-order-lag case (DGL/1) is summarized in Algorithm 7.

Algorithm 7 Differential Game Guidance (DGL/1).

1: Obtain the acceleration bounds $a_M^{\max}$, $a_T^{\max}$ and the critical time-to-go threshold
2: **for** each guidance cycle **do**
3: Read states; estimate the ZEM and the time-to-go t_{go}
4: Evaluate the capture zone (feasibility of a guaranteed hit)
5: Compute the bang-bang command $a_M^* = a_M^{\max}\,\mathrm{sgn}(\mathrm{ZEM})$
6: Enforce the DGL/1 switching rule: do not reverse the command after the critical t_{go} threshold
7: Send the command to the autopilot
8: **end for**

8 Learning-Based Guidance

Reinforcement learning (RL) formulates the guidance problem as a Markov decision process (MDP) [16, 17]. An MDP is defined by a state space $\mathcal{S}$, an action space $\mathcal{A}$, transition dynamics $p(s'|s,a)$, a reward $r(s,a)$, and a discount factor γ. The goal is to find the policy that maximizes the expected cumulative discounted reward:

$$J = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t\, r(s_t, a_t)\right]. \tag{31}$$

The action-value function (the Q-function), which gives the value of a state-action pair, satisfies the recursive Bellman equation:

$$Q^\pi(s,a) = \mathbb{E}\big[r(s,a) + \gamma\, Q^\pi(s',a')\big]. \tag{32}$$

This equation is the discrete, data-driven counterpart of the Hamilton-Jacobi-Bellman logic used in GENEX.

8.1 Fundamental Concepts

RL methods differ along two axes. If the policy is deterministic ($\pi(s) = a$) the same state always maps to the same action, and external noise is added for exploration; if it is stochastic

($\pi(a|s)$) a probability distribution is produced and the action is sampled, which provides exploration intrinsically. If learning is on-policy, only data generated by the current policy is used and old data is discarded; if it is off-policy, data from past policies is stored in a replay buffer and reused. For continuous acceleration commands an actor-critic structure is generally used: the actor learns the policy and the critic learns the Q-function.

8.2 Mapping to Guidance

In a typical design the state space contains the LOS rate, range, closing velocity, ZEM, and angular rate; the action space contains the acceleration command (or the distribution ratio between aerodynamic surfaces and thrust vector control); and the reward function is a weighted sum of the miss distance, control effort, impact angle, and field-of-view penalties. A common approach is to place RL on top of a classical law rather than generating the command from scratch (a residual or hybrid design), which preserves the reliability of the classical law [16].

8.3 Deep RL Algorithms

Four deep RL algorithms are widely used for continuous control. All four are actor-critic but differ in policy type (deterministic or stochastic) and in data usage (on-policy or off-policy). Each algorithm is described below together with its training loop.

8.3.1 DDPG

DDPG (Deep Deterministic Policy Gradient) is an off-policy actor-critic method with a deterministic policy. It reuses past data through a replay buffer, stabilizes learning with target networks, and adds exploration noise to the deterministic actor. It is sample-efficient but sensitive to hyperparameters and prone to overestimating the Q-values.

Algorithm 8 DDPG (Deep Deterministic Policy Gradient). Type: deterministic policy, off-policy.

Initialize actor μ_ζ and critic Q_χ; copy target networks $\zeta' \leftarrow \zeta$, $\chi' \leftarrow \chi$; empty replay buffer $\mathcal{D}$
for each time step **do**
    Observe state s_t; select action $a_t = \mu_\zeta(s_t) + \mathcal{N}_t$ (exploration noise)
    Apply a_t, observe r_t and s_{t+1}; store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{D}$
    Sample a random minibatch from $\mathcal{D}$
    Critic target: $y = r + \gamma\, Q_{\chi'}(s', \mu_{\zeta'}(s'))$; update Q_χ by minimizing $(y - Q_\chi)^2$
    Update actor by the policy gradient $\nabla_\zeta J \approx \mathbb{E}[\nabla_a Q\, \nabla_\zeta \mu]$
    Soft-update targets: $\zeta' \leftarrow \tau\zeta + (1-\tau)\zeta'$, likewise χ'
end for

8.3.2 TD3

TD3 (Twin Delayed DDPG) improves the stability of DDPG through three additions: twin critics whose minimum is used to reduce overestimation, delayed updates of the actor relative to the critics, and target policy smoothing that adds clipped noise to the target action. It remains deterministic and off-policy.

Algorithm 9 TD3 (Twin Delayed DDPG). Type: deterministic policy, off-policy.

Initialize two critics Q_{χ_1}, Q_{χ_2} and actor μ_ζ; copy target networks; empty $\mathcal{D}$
for each time step **do**
    Select $a_t = \mu_\zeta(s_t) + \mathcal{N}_t$; apply; store transition in $\mathcal{D}$
    Sample a minibatch; form smoothed target action $\tilde{a} = \mu_{\zeta'}(s') + \text{clip}(\epsilon)$
    Twin target (reduces overestimation): $y = r + \gamma \min_{i=1,2} Q_{\chi_i'}(s', \tilde{a})$
    Update both critics toward y
    Every d steps: update the actor and soft-update all target networks (delayed update)
end for

8.3.3 PPO

PPO (Proximal Policy Optimization) is an on-policy method with a stochastic policy. It updates the policy within a limited trust region using a clipped surrogate objective, which makes it very stable and easy to tune. Because it is on-policy the collected data is discarded after each update, so it is sample-inefficient.

Algorithm 10 PPO (Proximal Policy Optimization). Type: stochastic policy, on-policy.

Initialize stochastic policy π_θ and value network V_ϕ
for each iteration **do**
    Run the current policy in the environment and collect a batch of trajectories
    Estimate the advantage A_t (e.g. generalized advantage estimation) and the returns
    Define the probability ratio $\rho_t = \pi_\theta(a_t|s_t)/\pi_{\theta_{\text{old}}}(a_t|s_t)$
    For K epochs, maximize the clipped surrogate $L = \mathbb{E}[\min(\rho_t A_t,\ \text{clip}(\rho_t, 1-\varepsilon, 1+\varepsilon)A_t)]$
    Update the value network; discard the collected data
end for

8.3.4 SAC

SAC (Soft Actor-Critic) is an off-policy method with a stochastic policy and a maximum-entropy objective. Adding an entropy term to the reward encourages exploration and robustness. It uses twin critics as in TD3 and automatically tunes the entropy temperature, combining off-policy sample efficiency with strong exploration.

Algorithm 11 SAC (Soft Actor-Critic). Type: stochastic policy, off-policy, maximum entropy.

Initialize stochastic actor π_ζ, two critics Q_{χ_1}, Q_{χ_2}, and target networks; empty $\mathcal{D}$
for each time step **do**
    Sample $a_t \sim \pi_\zeta(\cdot|s_t)$; apply; store transition in $\mathcal{D}$
    Sample a minibatch; entropy-regularized target with $a' \sim \pi_\zeta(\cdot|s')$:
    $y = r + \gamma\big(\min_i Q_{\chi'_i}(s', a') - \alpha \log \pi_\zeta(a'|s')\big)$
    Update both critics toward y
    Update the actor to maximize $\mathbb{E}[\min_i Q_{\chi_i}(s, a) - \alpha \log \pi_\zeta(a|s)]$
    Auto-tune the temperature α toward a target entropy; soft-update target networks
end for

8.4 Algorithm Selection

DDPG is sample-efficient but brittle; TD3 reduces this brittleness through twin critics, delayed updates, and target smoothing. PPO is very stable and easy to tune but is sample-inefficient because it is on-policy. SAC combines off-policy efficiency with entropy-driven robustness and frequently gives the best performance on continuous-control benchmarks. No method is universally superior: when the simulator is cheap and fast the stability of PPO is attractive, whereas when data is expensive the sample efficiency of SAC or TD3 becomes decisive.

8.5 Training and Sim-to-Real

Training is carried out over a large number of simulated episodes. In curriculum learning the scenario difficulty is increased gradually; in domain randomization the training parameters are diversified to improve the robustness of the policy. The most important challenge is that a policy learned in simulation may not perform identically on the real system (the sim-to-real gap). This is mitigated by domain randomization, by observer-based structures that estimate uncertainty online [17], and by hybrid designs with classical laws.

9 Comparative Discussion

Table 1 summarizes the reviewed methods by their principal properties. Classical proportional navigation is simple and reliable but does not control the impact angle and struggles against maneuvering targets. GENEX adds impact-angle control as an optimal generalization but relies on target and model information. SMC emphasizes robustness; DGL provides a worst-case guarantee but at a high computational load. Learning-based methods offer flexibility but are weak in terms of theoretical guarantees and interpretability.

Table 1: Comparative summary of terminal guidance methods.

Method	**Core idea**	**Strength / weakness**	**Suitable scenario**
PN family	Null the LOS rate	Simple, reliable / no angle control, weak in maneuver	General terminal guidance
GENEX	Miss plus impact angle, optimal	Angle control, flexible profile / needs model info	Impact-angle-constrained missions
SMC	LOS rate as sliding surface	Robust, no estimation needed / chattering	Maneuvering target, uncertainty
DGL	Target as intelligent adversary	Worst-case guarantee / high computation	Agile target, missile-versus-missile
RL	Learn the policy from data	Flexible, adaptive / no guarantee, black box	Highly constrained, nonlinear problems

There are strong conceptual links among the methods. Equations (3), (8), and (13) reveal the common ZEM basis of PN, APN, and GENEX. Equations (27) and (32) show that differential game guidance and reinforcement learning are the continuous and discrete expressions of the same dynamic-programming principle. These links explain why the current trend moves toward hybrid designs that combine the guarantees of classical laws with the flexibility of learning-based methods.

10 Conclusions

This survey presented terminal guidance laws, from classical proportional navigation to learning-based approaches, within a common framework. The main conclusions are as follows.

1. Classical laws provide analytic reliability and low computational load, and the proportional navigation family is unified by the zero-effort-miss form in (3).

2. Optimal and explicit methods, in particular GENEX, control additional constraints such as the impact angle, and reduce to PN and APN as special cases.

3. Robust and game-theoretic methods (SMC and DGL) provide guarantees against uncertainty and against intelligent targets, at the cost of chattering or high computation respectively.

4. Learning-based methods offer flexibility in high-dimensional and nonlinear problems but lack theoretical guarantees and interpretability.

5. No method is superior in every scenario; the choice depends on the target behaviour, the constraints, the available information, and the computational budget. Given the shared zero-effort-miss and dynamic-programming structure, hybrid designs that combine classical guarantees with learning-based flexibility are the most promising future direction.

References

[1] E. J. Ohlmeyer and C. A. Phillips, "Generalized vector explicit guidance," *Journal of Guidance, Control, and Dynamics*, vol. 29, no. 2, pp. 261-268, 2006.

[2] J. H. Seo, *Study on Optimal and Sliding Mode Guidance for an Interception Problem*. M.S. thesis, Oklahoma State University, 2000.

[3] P. Zarchan, *Tactical and Strategic Missile Guidance*. AIAA Progress in Astronautics and Aeronautics.

[4] N. A. Shneydor, *Missile Guidance and Pursuit: Kinematics, Dynamics and Control*. Horwood Publishing, 1998.

[5] J. Moon, K. Kim, and Y. Kim, "Design of missile guidance law via variable structure control," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 4, 2001.

[6] K. R. Babu, I. G. Sarma, and K. N. Swamy, "Switched bias proportional navigation for homing guidance against highly maneuvering targets," *Journal of Guidance, Control, and Dynamics*, 1994.

[7] J. I. Lee, I. S. Jeon, and M. J. Tahk, "Guidance law to control impact time and angle," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 301-310, 2007.

[8] K. S. Erer and R. Tekin, "Impact time and angle control based on constrained optimal solutions," *Journal of Guidance, Control, and Dynamics*, 2016.

[9] S. R. Kumar, S. Rao, and D. Ghose, "Sliding-mode guidance and control for all-aspect interceptors with terminal angle constraints," *Journal of Guidance, Control, and Dynamics*, 2012.

[10] N. Harl and S. N. Balakrishnan, "Impact time and angle guidance with sliding mode control," *IEEE Transactions on Control Systems Technology*, 2011.

[11] B. Kada, "Arbitrary-order sliding-mode-based homing-missile guidance for intercepting highly maneuverable targets," *Journal of Guidance, Control, and Dynamics*, 2014.

[12] Y. Shtessel and C. Edwards, "Sliding mode control in aerospace applications: A survey," *International Journal of Robust and Nonlinear Control*, 2026.

[13] J. Shinar and T. Shima, "Nonorthodox guidance law development approach for intercepting maneuvering targets," *Journal of Guidance, Control, and Dynamics*, 2002.

[14] V. Turetsky and J. Shinar, "Missile guidance laws based on pursuit-evasion game formulations," *Automatica*, 2003.

[15] I. E. Weintraub, M. Pachter, and E. Garcia, "An introduction to pursuit-evasion differential games," *American Control Conference*, 2020.

[16] J. Park and C. K. Ryoo, "Enhancing terminal missile capturability via progressive training and deep deterministic policy gradient-based reinforcement learning," *IEEE Access*, 2026.

[17] W. Wang and Z. Chen, "Observer-based deep reinforcement learning for robust missile guidance and control," *IEEE Access*, 2025.

[18] X. Guo and W. Ma, "Research status and prospects of constrained cooperative guidance: A review," *Actuators*, 2026.

[19] D. Engelsman and I. Klein, "Guidance, navigation, and control: A survey, taxonomy, and challenges (2020-2025)," 2025.