# Technical Appendix - VTOL Precision Landing Simulator

## 1) Notation (units)

z: altitude above landing pad (m).
W: camera image width in pixels (px). H: height in px. HFOV: horizontal field of view (deg).
f_px: focal length in pixels. S: marker physical size (m). px: marker pixel span (px).
x, y: lateral pad-relative coordinates (m). dt: time step (s), default dt = 1.
q: process noise scaling for the constant-velocity model. sigma: measurement std dev (m).

## 2) Camera geometry (pinhole, nadir)

Focal length in pixels: f_px = W / (2 * tan(HFOV / 2)).
Marker pixel size model: px = f_px * S / max(z, eps). We use eps = 1e-6 to avoid division by zero.
Field-of-view gate (approx.): radial <= z * tan(HFOV / 2), where radial = sqrt(x^2 + y^2).

## 3) Vision detection and lock

Base detection probability (logistic): base = 1 / (1 + exp(-k * (px - thresh_px))). We use k = 0.25.
Modifiers: light_boost = 0.6 + 0.4 * illum; blur_penalty = (1 - 0.6 * blur); backend_boost: 1.0 for ArUco, 1.1 for AprilTag.
Final probability: p_det = clip(base * light_boost * blur_penalty * backend_boost, 0, 1).
Dwell and lock logic: detected -> dwell += 1, else dwell = 0. If dwell >= N, lock = True.
Unlock: if locked and i > 1/3 of the approach, unlock with small probability p ~ 0.05 when not detected.

## 4) Beacon correction (when locked)

When locked, pull raw XY toward the pad center (beacon-like correction):
pos_raw = pos_raw + (-g * pos_raw), where g in [0.0, 0.8].

## 5) Constant-velocity Kalman filter (2D)

State: X = [x, y, vx, vy]^T. Measurement: z_meas = [x, y]^T.
Dynamics (dt = 1): A = [[1,0,1,0],[0,1,0,1],[0,0,1,0],[0,0,0,1]]. H = [[1,0,0,0],[0,1,0,0]].
Process noise: Q = q * [[1/4,0,1/2,0],[0,1/4,0,1/2],[1/2,0,1,0],[0,1/2,0,1]].
Measurement noise: R = diag([sigma^2, sigma^2]).
Predict: X = A X; P = A P A^T + Q. Update: K = P H^T (H P H^T + R)^{-1}; X = X + K (z - H X); P = (I - K H) P.

# Technical Appendix (continued)

## 6) Measurement noise scheduling (R)

Unlocked: sigma = kf_r_base (GPS ~1.0 m; RTK ~0.03 to 0.10 m).
Locked: sigma = clip( 0.8 / max(px_est, 1.0), 0.02, 0.20 ).
Interpretation: larger marker in image -> smaller sigma -> tighter R -> more measurement trust.

## 7) Landing cone and corridor

Allowed radius: r_allowed = (z / z_top) * r_top. In app: z_top = 10 m, r_top = 1.0 m.
Cone violation per frame: radial > r_allowed. Violation rate is mean over the run.

## 8) Metrics

Pad accuracy (XY): e_xy = sqrt(x_TD^2 + y_TD^2) using final Kalman position.
Touchdown vertical speed: v_z = max(0, (z[-k-1] - z[-1]) / (k * dt)), e.g., k = 5.
Cone violation rate: mean(radial > r_allowed). Lock stability: mean(locked over last 30 percent of frames).

## 9) Scoring (0 to 100)

score = 100 * ( 0.40 * exp(-e_xy / 0.20) + 0.20 * exp(-max(0, v_z - 0.5) / 0.5) + 0.20 * exp(-5 * viol_rate) + 0.20 * lock_stability ).
Goal: encourage pad accuracy and soft touchdown, reward cone compliance and stable final lock.

## 10) Numerical stability and reproducibility

Use eps = 1e-6 in denominators; clip probabilities to [0,1]; clip sigma to [0.02,0.20] when locked.
Set a random seed for reproducible tuning; keep dt consistent across A, Q, and v_z computation.

## 11) Parameter ranges (typical)

RTK sigma: 0.02..0.10 m; GPS sigma: 0.5..2.0 m.
Lock threshold: 20..40 px; dwell: 6..12 frames; marker size: 0.3..0.8 m.
Kalman q: ~3e-5..1e-2; beacon gain g: 0.2..0.6.

## 12) Data export schema (CSV)

t (frame), x_raw (m), y_raw (m), x_kf (m), y_kf (m), z_agl (m), detected (0/1), locked (0/1), px_est (px).