SAM (地対空ミサイル) システム Lua スクリプト仕様書

1. 概要

このドキュメントは、Stormworks で開発中の SAM (Surface-to-Air Missile) システムを制御する 2 つの Lua スクリプト、`KalmanFilter_SAM.lua` と `GuidanceController2.lua` の仕様について記述します。

- **`KalmanFilter_SAM.lua`:** ミサイルに搭載されたレーダーからの観測値と、発射母機からのデータリンク情報を統合し、拡張カルマンフィルター (EKF) を用いて目標のグローバル座標系における位置と速度を推定します。
- **`GuidanceController2.lua`:** `KalmanFilter_SAM.lua` からの目標情報とミサイル自身のセンサー情報に基づき、誘導計算（初期誘導：単純追尾、通常誘導：比例航法）を行い、フィン制御信号と近接信管信号を出力します。

これら 2 つのスクリプトは連携して動作し、目標への誘導を実現します。

2. `KalmanFilter_SAM.lua` (v0.4 相当)

2.1. 機能

- ミサイル搭載レーダーからの目標観測値（最大 6 目標）を処理します。
- データリンクからの目標座標情報を受け取ります。
- データリンク座標または追跡中の目標の予測位置を基準として、レーダー観測値とのデータアソシエーション（対応付け）を行います。
    - トラックが存在しない場合、またはデータリンク要求フラグがオンの場合、データリンク座標に最も近い観測値でフィルターを初期化します。
    - トラックが存在する場合、予測位置に対して EKF 更新を試算し、誤差指標 (ε) が最小かつ閾値以下の観測値を対応付けます。
- 拡張カルマンフィルター (EKF) を使用して、対応付けられた観測値に基づき目標の状態（グローバル位置および速度）を推定・更新します。状態ベクトルは $[x, v_x, y, v_y, z, v_z]^T$ です。
- 一定時間観測が途絶えた目標の追跡を中断（ロスト処理）します。
- 推定した目標の状態や、`GuidanceController2.lua` で必要となる各種情報をパススルーして出力します。

2.2. 入力 (コンポジット信号)

- **オンオフ:**
    - `1`: データリンク要求フラグ (追跡中の目標と有効なデータリンクがかけ離れている場合に受信)
    - `2`: レーダー有効範囲フラグ (目標がレーダーの有効範囲内にいるか)
    - `3`: 発射フラグ (ミサイルが発射されたか)
    - `4`: データリンク更新停止フラグ
- **数値:**
    - `1-4`: レーダー目標 1 (距離[m], 方位角[回転単位], 仰角[回転単位], 経過時間[Tick])
    - `5-8`: レーダー目標 2 ...
    - `9-12`: レーダー目標 3 ...

    - `13-16`: レーダー目標 4 ...
    - `17-20`: レーダー目標 5 ...
    - `21-24`: レーダー目標 6 ...
    - `8`: 自機グローバル位置 X [m]
    - `12`: 自機グローバル位置 Y [m]
    - `16`: 自機グローバル位置 Z [m]
    - `20`: 自機オイラー角 Pitch [rad]
    - `24`: 自機オイラー角 Yaw [rad]
    - `25`: 自機オイラー角 Roll [rad]
    - `26`: データリンク目標座標 X [m]
    - `27`: データリンク目標座標 Y [m]
    - `28`: データリンク目標座標 Z [m]
    - `29`: (外部計算) データリンク目標との距離の 2 乗 [m^2]
    - `30`: 自機 X 軸速度 (ローカル) [m/s] (パススルー用)
    - `31`: 自機 Y 軸速度 (ローカル) [m/s] (パススルー用)
    - `32`: 自機 Z 軸速度 (ローカル) [m/s] (パススルー用)

2.3. 出力 (コンポジット信号)

- **オンオフ:**
    - `1`: トラッキング成功フラグ (`isTracking`) - EKF が有効な観測で更新された場合に True
    - `2`: 発射フラグ (入力 ch3 のパススルー)
- **数値:**
    - `1`: 推定目標座標 X [m] (Logic Delay 考慮済み)
    - `2`: 推定目標座標 Y [m] (Logic Delay 考慮済み)
    - `3`: 推定目標座標 Z [m] (Logic Delay 考慮済み)
    - `4`: 推定目標速度 Vx [m/s]
    - `5`: 推定目標速度 Vy [m/s]
    - `6`: 推定目標速度 Vz [m/s]
    - `7`: データリンク座標 X [m] (入力 ch26 のパススルー)
    - `8`: データリンク座標 Y [m] (入力 ch27 のパススルー)
    - `9`: データリンク座標 Z [m] (入力 ch28 のパススルー)
    - `10`: 自機グローバル位置 X [m] (入力 ch8 のパススルー)
    - `11`: 自機グローバル位置 Y [m] (入力 ch12 のパススルー)
    - `12`: 自機グローバル位置 Z [m] (入力 ch16 のパススルー)
    - `13`: 自機オイラー角 Pitch [rad] (入力 ch20 のパススルー)
    - `14`: 自機オイラー角 Yaw [rad] (入力 ch24 のパススルー)
    - `15`: 自機オイラー角 Roll [rad] (入力 ch25 のパススルー)
    - `16`: 自機 X 軸速度 (ローカル) [m/s] (入力 ch30 のパススルー)
    - `17`: 自機 Y 軸速度 (ローカル) [m/s] (入力 ch31 のパススルー)
    - `18`: 自機 Z 軸速度 (ローカル) [m/s] (入力 ch32 のパススルー)
    - `32`: 最新のイプシロン ε (誤差指標)

2.4. プロパティ

- `D_ASOC_EPS`: データアソシエーションで使用する誤差指標 ε の最大許容値。
- ~~`T_LOST`: 目標が更新されない場合にロストと判断するまでの Tick 数。~~

- `INIT_MAX_DIST`: フィルター初期化時に、データリンク座標とレーダー観測値がこの距離 [m] 以上離れていたら初期化しない。
- `P_BASE`: プロセスノイズの基本係数。
- `P_ADPT`: プロセスノイズの適応的スケーリング係数。
- `P_NOISE_EPS_THRS`: プロセスノイズの適応調整を開始する ε の閾値。
- `P_NOISE_EPS_SLOPE`: プロセスノイズの適応調整の傾き（ε に対する変化率）。
- `PRED_UNCERTAINTY_FACT`: 予測誤差共分散の不確かさ増加係数の底（時間経過に対する予測信頼度の低下率）。
- `LOGIC_DELAY`: 推定位置を出力する際の遅延補償 Tick 数。

2.5. 主要ロジック

1. **入力読み込み:** 各種センサー情報、データリンク情報、レーダー情報を読み込みます。
2. **レーダー観測処理:** レーダーが有効範囲内 (`isRadarEffectiveRange`) の場合、最大 `MAX_RADAR_TARGETS` 個の観測データを読み込み、`localToGlobalCoords` を用いてローカル極座標からグローバル直交座標・グローバル角度（方位角・仰角）に変換し、`currentObservations` テーブルに格納します。
3. **データアソシエーションと EKF:**
    - レーダー探知があり、観測データが存在する場合に実行します。
    - `trackedTarget` が存在しない、またはデータリンク再初期化要求 (`isDataLinkInitRequired`) がある場合：データリンク座標に最も近い観測データ (`bestInitObs`) を探し、`INIT_MAX_DISTANCE` 内であれば `initializeFilterState` でフィルターを初期化します。
    - `trackedTarget` が存在する場合：現在の状態から次の Tick の状態を予測 (`X_assoc_pred`) し、全ての `currentObservations` で EKF 更新を試算して誤差指標 ε を計算します。ε が最小かつ `DATA_ASSOCIATION_EPSILON_THRESHOLD` 以下の観測データ (`associatedObservation`) を選択します。
    - 最良のマッチが見つかった場合：`extendedKalmanFilterUpdate` を実行して `trackedTarget` の状態 (X, P, epsilon, lastTick, lastSeenTick) を更新し、`isTracking` を true にします。更新に失敗した場合はトラックを破棄します。
    - アソシエーションに失敗した場合は `isTracking` を false にします。
4. **目標ロスト判定:** ロスト判定の実装は削除しました。 ~~`isTracking` が false で、かつ最後に観測が紐付けられてから (`lastSeenTick`) 一定時間 (`TARGET_LOST_THRESHOLD_TICKS`) が経過した場合、`trackedTarget` を `nil` にして追跡を中断します。ロストするまでは予測ステップのみ実行して状態を維持します。~~
5. **出力処理:**
    - `isTracking` フラグを出力します。
    - `trackedTarget` が存在する場合、`LOGIC_DELAY` Tick 分の未来位置を予測して出力チャンネル 1-3 に、現在の推定速度をチャンネル 4-6 に、最新の ε をチャンネル 32 に出力します。
    - トラックがない場合は対応するチャンネルに 0 を出力します。
    - `GuidanceController2.lua` で必要な各種入力データ（データリンク座標、自機位置・姿勢・速度）をそのままパススルー出力します (ch7-18)。

2.6. 前提・注意点

- **座標系:** Stormworks の Physics Sensor 座標系 (X:東, Y:上, Z:北, 左手系) を基準とします。
- **`inv` 関数の実装:** 行列の逆行列を計算する `inv` 関数が別途実装されている必要があります。

- **EKF パラメータ:** 各種 EKF パラメータはプロパティから設定する必要があり、目標の特性やセンサー精度に合わせて調整が必要です。
- **初期化:** データリンク要求フラグやデータリンク更新停止フラグに基づいた初期化ロジックが追加されています。

3. `GuidanceController2.lua` (v0.2 相当)

3.1. 機能

- `KalmanFilter_SAM.lua` からの目標情報（推定位置・速度）とミサイル自身のセンサー情報（位置・速度・姿勢）を入力として受け取ります。
- 発射後の初期誘導フェーズ（一定時間）では、データリンク座標に基づいた**単純追尾**を行います。
- 通常誘導フェーズでは、**比例航法 (Proportional Navigation, PN)** を使用して目標を追尾します。
    - 目標情報のソースは、Kalman Filter の追跡状況 (`isTracking` フラグと `targetEpsilon`) に応じて切り替えます（フィルター有効時は推定値、無効時はデータリンク座標と差分速度）。
    - 比例航法の加速度指令は、$A_{cmd} = N \times V_c \times (\Omega \times \hat{R})$ に基づいて計算されます。
- ミサイル自身の姿勢を**クォータニオン**で管理します。
- 計算されたローカル座標系での加速度指令に基づき、ヨーおよびピッチ方向のフィン制御信号を出力します。
- 終末誘導フェーズにおいて、目標との推定距離と接近速度に基づき**近接信管**の起動信号を出力します。

3.2. 入力 (コンポジット信号)

- **オンオフ:**
    - 1: トラッキング成功フラグ (`isTracking` from KalmanFilter_SAM)
    - 2: 発射フラグ (`isLaunch` from external source or KalmanFilter_SAM)
- **数値:**
    - 1: 推定目標座標 X (`targetPosX` from KalmanFilter_SAM)
    - 2: 推定目標座標 Y (`targetPosY` from KalmanFilter_SAM)
    - 3: 推定目標座標 Z (`targetPosZ` from KalmanFilter_SAM)
    - 4: 推定目標速度 Vx (`targetVelX` from KalmanFilter_SAM)
    - 5: 推定目標速度 Vy (`targetVelY` from KalmanFilter_SAM)
    - 6: 推定目標速度 Vz (`targetVelZ` from KalmanFilter_SAM)
    - 7: データリンク目標座標 X (`dataLinkX` from KalmanFilter_SAM)
    - 8: データリンク目標座標 Y (`dataLinkY` from KalmanFilter_SAM, 下限 100m)
    - 9: データリンク目標座標 Z (`dataLinkZ` from KalmanFilter_SAM)
    - 10: 自機グローバル位置 X (`ownPosX` from KalmanFilter_SAM)
    - 11: 自機グローバル位置 Y (`ownPosY` from KalmanFilter_SAM)
    - 12: 自機グローバル位置 Z (`ownPosZ` from KalmanFilter_SAM)
    - 13: 自機オイラー角 Pitch (`ownPitch` from KalmanFilter_SAM)
    - 14: 自機オイラー角 Yaw (`ownYaw` from KalmanFilter_SAM)
    - 15: 自機オイラー角 Roll (`ownRoll` from KalmanFilter_SAM)
    - 16: 自機ローカル速度 Vx (`ownVelLocalX` from KalmanFilter_SAM)
    - 17: 自機ローカル速度 Vy (`ownVelLocalY` from KalmanFilter_SAM)
    - 18: 自機ローカル速度 Vz (`ownVelLocalZ` from KalmanFilter_SAM)

- `32`: イプシロン ε (`targetEpsilon` from KalmanFilter_SAM)

3.3. 出力 (コンポジット信号)

- **オンオフ:**
  - `1`: 近接信管起動信号 (`proximityFuseTrigger`)
- **数値:**
  - `1`: ヨー制御信号 (`yawControl`) - プラスが右旋回
  - `2`: ピッチ制御信号 (`pitchControl`) - プラスがピッチアップ

3.4. プロパティ

- `N`: 航法ゲイン (比例航法用)。
- `FinGain`: フィン制御ゲイン（比例航法におけるローカル加速度指令からフィン指令値への変換係数）。
- `SimpleTrackGain`: 単純追尾ゲイン（初期誘導における角度偏差からフィン指令値への変換係数）。
- `ProximityDistance`: 近接信管の作動距離 [m]。
- `MaxControl`: フィン制御出力の最大/最小値（例: 1.0 で ±1 に制限）。
- `PN_GUIDANCE_EPSILON_THRESHOLD`: 比例航法で Kalman Filter の推定値を使用するための ε の閾値。これより大きい場合はデータリンク情報を使用。
- `INITIAL_GUIDANCE_DURATION_SECONDS`: 発射後に初期誘導（単純追尾）を行う時間 [秒]。
- `FUSE_PROM_DURATION_TICKS`: 近接信管の距離予測に使用する時間 [Tick]。
- `GUIDANCE_START_ALTITUDE`: ミサイルがこの高度 [m] を超えたら誘導を開始する。
- `VT_FUSE_SAMPLE_TICKS`: 近接信管の接近速度計算に使用する過去データの Tick 数。

3.5. 主要ロジック

1. **起動と Tick カウント:** 発射フラグ (`isLaunch`) が True になったら Tick カウント (`launchTickCounter`) を開始します。
2. **入力読み取り:** `KalmanFilter_SAM` からのパススルー情報を含む各種データを読み込みます。データリンク目標 Y 座標には下限値 (100m) が設定されます。
3. **誘導開始判定:** 自機高度 (`ownPosY`) が `GUIDANCE_START_ALTITUDE` を超えたら `guidanceStart` フラグを True にし、誘導計算を開始します。
4. **姿勢管理:** 自機オイラー角から `eulerZYX_to_quaternion` を用いて現在の姿勢クォータニオン `ownOrientation` を計算します。
5. **変数準備:** 自機位置、目標位置、目標速度などをベクトル形式で準備します。自機速度はローカル座標で読み取り、`rotateVectorByQuaternion` を使ってグローバル座標 `ownVelVec` に変換します。データリンク座標の差分からデータリンク速度 `dataLinkVelVec` も計算します（通常誘導の PN では現在未使用）。
6. **誘導モード分岐:** `launchTickCounter` が `INITIAL_GUIDANCE_TICKS` 未満かどうかで処理を分岐します。
   - **初期誘導フェーズ (`launchTickCounter < INITIAL_GUIDANCE_TICKS`):**
     - データリンク座標 `dataLinkTargetPosVec` を目標とします。
     - `globalToLocalCoords` で目標のローカル座標 `targetLocal` を計算します。
     - `math.atan` を用いてローカルな方位角・仰角を計算し、`SIMPLE_TRACK_GAIN` を掛けて `yawControl`, `pitchControl` を決定します（単純追尾）。
     - データリンクが無効な場合は直進します。
     - 近接信管は作動しません。

- **通常誘導フェーズ (`launchTickCounter >= INITIAL_GUIDANCE_TICKS`):**
  - Kalman Filter が有効か (`isTracking` と `targetEpsilon`)、または目標との距離が近いか (`distanceSq < 500^2`) によって、比例航法に使用する目標情報 (`currentTargetPosVec`, `currentTargetVelVec`) を選択します。
    - 有効な場合: Kalman Filter の推定値を使用します。
    - 無効な場合: データリンク座標とデータリンク差分速度 (`dataLinkVelVec`) を使用します。（コードでは `currentTargetVelVec = dataLinkVelVec` となっています）
  - 選択された目標情報が存在する場合、比例航法計算を実行します。
    - 相対位置 R_{vec}、相対速度 V_{vec}、距離 R_{mag} を計算します。
    - 視線単位ベクトル $\hat{R}$、接近速度 V_c、LOS レートベクトル Ω を計算します。
    - 外積 $\Omega \times \hat{R}$ を計算します。
    - グローバル加速度指令 $A_{cmd_global} = N \times V_c \times (\Omega \times \hat{R})$ を計算します。
    - `rotateVectorByInverseQuaternion` でローカル加速度指令 A_{cmd_local} に変換します。
    - A_{cmd_local} の X, Y 成分と `FIN_GAIN` から `yawControl`, `pitchControl` を決定します。
  - 近接信管判定: Kalman Filter 有効時、または近距離強制終末誘導時に実行。`VcAverageCalculator` で計算した平均接近速度に基づき、`FUSE_PROXIMITY_DURATION_TICKS` 後の予測距離が `PROXIMITY_DISTANCE` 未満なら `proximityFuseTrigger` を True にします。

7. **制御量制限:** 計算された `yawControl`, `pitchControl` を `MAX_CONTROL` で制限します。
8. **出力:** `proximityFuseTrigger`, `yawControl`, `pitchControl` を出力します。

3.6. 前提・注意点

- **ヘルパー関数:** ベクトル演算、クォータニオン演算、座標変換関数が別途定義されている必要があります。
- **ゲイン調整:** `NAVIGATION_GAIN`, `FIN_GAIN`, `SIMPLE_TRACK_GAIN` はミサイルの特性に合わせて調整が必要です。
- **データリンク無効時:** 通常誘導フェーズで Kalman Filter が無効かつデータリンクも無効な場合の PN 計算の挙動（目標速度をどう扱うか）は注意が必要です。
- **近接信管:** 平均接近速度 `VcAverageCalculator` を使用した距離予測に基づいています。

4. 座標系

- **グローバル座標系:** Stormworks Physics Sensor 座標系 (左手系: +X:東, +Y:上, +Z:北) を基準とします。Kalman Filter の状態推定や比例航法の A_{cmd_global} はこの座標系です。
- **ローカル座標系:** ミサイル本体の座標系 (左手系: +X:右, +Y:上, +Z:前) を基準とします。Physics Sensor の速度入力や、フィン制御のための A_{cmd_local} はこの座標系です。