
```

function [xhatkp1, Pkp1] = ukf_step(dyn_fun, h_fun, dt, Q, R, alpha, beta,
kappa, z, xhat, Pxx)
%UKF Unscented Kalman Filter
% Initial constants
nx = length(xhat);
nv = size(Q, 1);
nz = size(R, 1);
na = nx + nv;
lambda = (alpha^2)*(na + kappa) - na;
npts = 2*na+1;

% Augment state for propagation
xa = [xhat; zeros(nv, 1)];
Pa = zeros(na, na);
Pa(1:nx, 1:nx) = Pxx;
Pa(end-nv+1:end, end-nv+1:end) = Q;

% Create sigma points
Sx = chol(Pa).';
chi = zeros(na, npts);
chi(:, 1) = xa;
for i=2:2:npts
    chi(:, i) = xa + sqrt(na + lambda)*Sx(:, i/2);
    chi(:, i+1) = xa - sqrt(na + lambda)*Sx(:, i/2);
end

% Propagate the points to the time of the meas (should accomodate meas
timestamp)
% Could use the same chi here, don't need the old points again
chi_transf = zeros(na, npts);
for i=1:npts
    [~, x] = ode113(dyn_fun, [0, dt], chi(:, i));
    chi_transf(:, i) = x(end, :).';
end
% Compute xbar and Pbar
w0m = lambda/(na+lambda);
w0c = lambda/(na+lambda) + 1 - alpha^2 + beta;
wi = 1/(2*(na+lambda));

xabar = w0m*chi_transf(:, 1) + sum(wi*chi_transf(:, 2:end), 2);
Pabar = w0c*(chi_transf(:, 1) - xabar)*(chi_transf(:, 1) - xabar).';
for i=2:npts
    Pabar = Pabar + wi*(chi_transf(:, i) - xabar)*(chi_transf(:, i) - xabar).';
end
xbar = xabar(1:nx);
Pbar = Pabar(1:nx, 1:nx);

% Measurement update
% Recompute na, npts, lambda, etc
na = nx+nz;
npts = 2*na+1;

```

```

lambda = (alpha^2)*(na+kappa) - na;
% Augment to new xa Pa
xa = [xbar; zeros(nz, 1)];
Pa = zeros(na, na);
Pa(1:nx, 1:nx) = Pbar;
Pa(end-nz+1:end, end-nz+1:end) = R;

% Form new sigma points
Sx = chol(Pa).';
gamma = zeros(na, npts);
gamma(:, 1) = xa;
for i=2:2:npts
    gamma(:, i) = xa + sqrt(na + lambda)*Sx(:, i/2);
    gamma(:, i+1) = xa - sqrt(na + lambda)*Sx(:, i/2);
end

% Transform points through the measurement function (state-> meas space)
% Could use the same gamma here, don't need the old points again
gamma_transf = zeros(nz, npts);
for i=1:npts
    gamma_transf(:, i) = h_fun(gamma(1:nx, i), gamma(nx+1:end, i)); % One
bearing is on the order of the range
end
% Compute xbar and Pbar
w0m = lambda/(na+lambda);
w0c = lambda/(na+lambda) + 1 - alpha^2 + beta;
wi = 1/(2*(na+lambda));

zbar = w0m*gamma_transf(:, 1) + sum(wi*gamma_transf(:, 2:end), 2); % Very
reasonable
Pzz = w0c*(gamma_transf(:, 1) - zbar)*(gamma_transf(:, 1) - zbar).';
for i=2:npts
    Pzz = Pzz + wi*(gamma_transf(:, i) - zbar)*(gamma_transf(:, i) - zbar).';
end
Pxz = w0c*(gamma(1:nx, 1) - xbar)*(gamma_transf(:, 1) - zbar).';
for i=2:npts
    Pxz = Pxz + wi*(gamma(1:nx, i) - xbar)*(gamma_transf(:, i) - zbar).';
end

% Compute the LMMSE update
xhatkpl = xbar + (Pxz/Pzz)*(z.' - zbar);
Pkpl = Pbar - Pxz*Pxz.';

end

```

Published with MATLAB® R2025a