Contents


```
function [cmd,cmdDot] = UAVGuidance( t, x, p )
```

Generate velocity, heading and altitude commands for a UAV

```
State:     x = [V;gama;psi;x;y;h;Tbar]
-------------------------------------
  V     true airspeed
  gama  air relative flight path angle
  psi   air relative flight heading angle
  x     East position
  y     North position
  h     altitude
  Tbar  normalized excess thrust

Command:   cmd = [v;psi;h;x;y]
------------------------------
  v       velocity command (true airspeed, m/s)
  psi     heading command (rad)
  h       altitude command (m)
  x       eastward position (m)
  y       northward position (m)
```

```
%------------------------------------------------------------------------
%   Form:
%   [cmd,cmdDot,p] = UAVGuidance( t, x, p );
%------------------------------------------------------------------------
%
%   ------
%   Inputs
%   ------
%   t   (1,1)     Current time
%   x   (7,1)     Current state vector
%   p   (.)       Flight parameters. This function uses the following
%                 fields:
%                   wp        (3,1)   Target waypoint position (x,y,h)
%                   Rmin      (1,1)   Minimum turn radius (m)
%                   hDotMax   (1,1)   Maximum climb rate (abs val)
%                   duration  (1,1)   Max duration to simulate (sec)
%                   dT        (1,1)   Time step
%                   stopSim = @(t,x)  Anonymous function. Sim terminates
%                                         when this evaluates to true.
%
%   -------
%   Outputs
%   -------
%   cmd     (5,1)     Commanded velocity, heading, altitude, and horizontal
%                       position.  [v;psi;h;x;y]
%   cmdDot  (3,1)     Commanded rate of change of velocity, heading, altitude.
%                       [vDot;psiDot;hDot]
%
%------------------------------------------------------------------------

% If no guidance parameters are provided, just keep flying along current
```

```
% trajectory
if isempty(p)
  cmd = x([1 3 6 4 5]); % current state values for: [v, psi, h, x, y]
  cmdDot = zeros(3,1);
  return
end

% Turning -- velocity and heading command
[vDotCmd,psiDotCmd] = UAVAutoTurn( x, p.wp, p.Rmin, p.dT );
vCmd = x(1)+vDotCmd*p.dT;
psiCmd = x(3)+psiDotCmd*p.dT;

% Follow the turn -- lateral position commands
xCmd = x(4) + vCmd*sin(psiCmd);
yCmd = x(5) + vCmd*cos(psiCmd);

% Climbing -- altitude commands
dh = p.wp(3) - x(6);
if abs(dh/p.dT)>p.hDotMax
  hDotCmd = p.hDotMax*sign(dh);
else
  hDotCmd = dh/p.dT;
end
hCmd = p.wp(3);


% Stack the commands into vectors
cmd = [vCmd;psiCmd;hCmd;xCmd;yCmd];
cmdDot = [vDotCmd;psiDotCmd;hDotCmd];
```

```
function [vDot,psiDot] = UAVAutoTurn( state, wp, Rmin, dT )
% getting varibales from input
x0 = state(4);
y0 = state(5);

xT = wp(1);
yT = wp(2);

v0 = state(1);
gamma0 = state(2);
psi0 = state(3);

% center of turning circle: potentially two sides
xC1 = x0+Rmin*cos(psi0);
yC1 = y0-Rmin*sin(psi0);

xC2 = x0+Rmin*cos(psi0+pi);
yC2 = y0-Rmin*sin(psi0+pi);

% choose the side closer to the target as long as it is > R
d1 = norm([xT-xC1;yT-yC1]);
d2 = norm([xT-xC2;yT-yC2]);

if d1<Rmin || d2<Rmin
  % fly straight for now
  psiDot = 0;
else

  % desired heading
```

```
  psiT = atan2(xT-x0,yT-y0);

  % est time to destination (assuming correct heading)
  xDot = v0*cos(gamma0)*sin(psi0);
  yDot = v0*cos(gamma0)*cos(psi0);

  tDest = sqrt((xT-x0)^2 + (yT-y0)^2) / sqrt(xDot^2 + yDot^2);

  if abs(psiT-psi0)>pi
    1;
  end

  while ( (psiT-psi0) < -pi )
    psiT = psiT+2*pi;
  end
  while ( (psiT-psi0) > pi )
    psiT = psiT-2*pi;
  end

  % psiDot = (psiT-psi0)/dT;
  % Turn slightly slower than literally dT
  psiDot = (psiT-psi0)/(.1*tDest);

  if abs(psiDot)>v0/Rmin
    psiDot = sign(psiDot)*v0/Rmin;
  end

end

vDot = 0;
```

Published with MATLAB® R2020a