```
function [theta_in, theta_exit, theta_vec, path, distance,
 vel_out, g_s] = transition(vel_in, pos_in, vel_exit, r, dir,h0, n,
 theta_in_ref, theta_exit_ref, theta_vec_ref)

   [theta_in, theta_exit] = transition_startstop_angles(vel_in,
 vel_exit, dir);

   if isnan(theta_in_ref)
       [path, distance, vel_out, theta_vec] =
 transition_path(vel_in,vel_exit, pos_in,theta_in, theta_exit, r, dir,
 h0, n);
   else
       [path, distance, vel_out, theta_vec] =
 transition_path(vel_in,vel_exit, pos_in,theta_in_ref, theta_exit_ref,
 r, dir, h0, n);
   end

   if isnan(theta_vec_ref)
       g_s = transition_g_s(vel_in, path, theta_vec, r, h0, n);
   else
       g_s = transition_g_s(vel_in, path, theta_vec_ref, r, h0, n);
   end
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% COMPLETE FUNCTIONS BELOW %%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%

function [theta_in, theta_exit] = transition_startstop_angles(vel_in,
 vel_exit, dir)

       % Assume always going to be bottom half of a circle
       % First, compute the necessary starting angle and stopping
 angle for the loop

       if dir == 1 % cw
           %split of cw into two seperate cases for initial angle
           if sign(vel_in(3)) ~= -1
               theta_in = pi - atan2(vel_in(3), vel_in(2)); % see
 instructional guide for derivation
           else
               theta_in = -(pi+atan2(vel_in(3), vel_in(2)));
           end
           theta_exit = pi - atan2(vel_exit(3), vel_exit(2));
       else %ccw
           theta_in = atan2(vel_in(3),vel_in(2));
           theta_exit = atan2(vel_exit(3), vel_exit(2));
       end
end
```

```matlab
function [path, distance, vel_out, theta_vec] =
 transition_path(vel_in, vel_exit, pos_in,theta_in, theta_exit, r,
 dir, h0, n)

    % create vector of theta values
    theta_vec = linspace(theta_in, theta_exit, n);

    g = 9.81;

    % Split into cw and ccw cases
    if dir == 1 %cw
        path = pos_in + [zeros(length(theta_vec),1), r*sin(theta_in)
 - r*cos(theta_vec' - pi/2), r*cos(theta_in) + r*sin(theta_vec' -
 pi/2)]; %[x_pos, y_pos, z_pos]
    else %ccw
        path = pos_in + [zeros(length(theta_vec),1), -r*sin(theta_in)
 + r*cos(theta_vec' - pi/2), r*cos(theta_in) + r*sin(theta_vec' -
 pi/2)]; %[x_pos, y_pos, z_pos]
    end

    % find distance along track element - use S = rtheta
    distance = r*(theta_vec'-theta_in);

    mag = sqrt(2*g*(h0 - path(end,3)));

    % use what we know about the exit velocity direction to compute
    if sign(vel_in(2)) > 0
        vel_out = [0, mag*cos(theta_vec(end)),
 mag*sin(theta_vec(end))];
    else
        vel_out = [0, -mag*cos(theta_vec(end)),
 mag*sin(theta_vec(end))];
    end
end


function g_s = transition_g_s(vel_in, path, theta_vec, r, h0, n)
% Compile the g forces into matrix, evaluated at each theta in
 theta_vec
    g = 9.81;

    mag = sqrt(2*g*(h0-path(:,3)));
    vx = vel_in(1);
    vy = vel_in(2);
    vz = vel_in(3);

    vel = [mag.*cos(theta_vec'), vy*ones(length(theta_vec),1),
 mag.*sin(theta_vec')];

    g_s = [zeros(length(theta_vec'),1), zeros(length(theta_vec),1),
 (mag.^2)/(r*g) - sin(theta_vec'- pi/2)];   %G-force matrix [front/
back, left/right, up/down]
end
```

```
Not enough input arguments.

Error in transition (line 3)
    [theta_in, theta_exit] = transition_startstop_angles(vel_in,
 vel_exit, dir);
```

Published with MATLAB® R2021a