```
f_vz = 1;   % sampling frequency

% parameters of satellite
v_el = 0.167;       % angular rate of elevation of the satellite
t_max = 180/v_el;   % time it takes for the satellite to end its journey
azim = 30.067;      % initial azimuth of the satellite

t = 0:1/f_vz:t_max; % time vector
tlen = length(t);
elev = horzcat(v_el*t(1:tlen/2), -v_el*t(tlen/2+1:end) + 180);      % vector of
elevations of the satellite in time

figure(1)
subplot(311), plot(t,elev)
title("Az 0-360°, El 0-90°")
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sat")

% capabilities of the tracking system
ant_az_step = 1;      % min. angle step in deg - azimuth
ant_el_step = 1;      % min. angle step in deg - elevation
ant_v_az    = 4.5;    % max. angular rate of azimuth of the antenna
ant_v_el    = 4.5;    % max. angular rate of elevation of the antenna

ant_t_az    = ant_az_step/ant_v_az;     % time it takes for the antenna to change
its azimuth direction
ant_t_el    = ant_el_step/ant_v_el;     % time it takes for the antenna to change
its elevation direction

ant_azel = zeros(length(t),2);  % instructions for the tracker motors
ant_azel(1,:) = [azim,0];       % set first position
ant_azel(:,1) = ant_az_step*round(azim/ant_az_step)*ones(length(t),1); % set all
azims to azim

simazel = zeros(length(t),2);   % only for vizualizing
simazel(1,:) = [azim,0];       % set first position
simazel(:,1) = ant_az_step*round(azim/ant_az_step)*ones(length(t),1); % set all
azims to azim
simazel_prev = 0;
ant_is_moving = false;
for i = 2:length(t)
    % current position of the sattelite - [azim,elev(t)]
    if and(i >= tlen/2 , i <= tlen/2 + 40/f_vz)
        ant_azel(i,2) = 90;
        simazel(i,2) = 90;
        ant_is_moving = false;
        simazel_prev = 90;
    elseif i >= tlen/2

    % down
        ant_azel(i,2) = ant_el_step*(round(elev(i)/ant_el_step));
    if(ant_is_moving == true)
        simazel(i,2) = simazel(i-1,2) - ant_v_el/f_vz;
        if(simazel(i,2) <= ant_el_step*(floor(elev(i)/ant_el_step)))
            ant_is_moving = false;
            simazel(i,2) = ant_el_step*(floor(elev(i)/ant_el_step));
             if i < 700, sprintf("END Moving DOWN %i", i),end
        end
        simazel_prev = simazel(i,2);
        continue;
    end

    if(ant_is_moving == false)
        simazel(i,2) = simazel_prev;

        future_el = elev( min(i+floor((ant_t_el/2)*f_vz),length(elev)) );
        motor_el = ant_el_step*(ceil(elev(i)/ant_el_step)) -
ant_v_el*(ant_t_el/2);
        if i < 650 * f_vz
                sprintf("%g, %g , %i", motor_el, future_el, i)
        end
        if(future_el <= motor_el)
            ant_is_moving = true;
            if i < 700
                sprintf("%g < %g => Moving DOWN %i", motor_el, future_el, i)
            end
        end
    end
    else
    % UP
    ant_azel(i,2) = ant_el_step*(round(elev(i)/ant_el_step));
    if(ant_is_moving == true)
        simazel(i,2) = simazel(i-1,2) + ant_v_el/f_vz;
        if(simazel(i,2) > ant_el_step*(floor(elev(i)/ant_el_step)+1))
            ant_is_moving = false;
            simazel(i,2) = ant_el_step*(floor(elev(i)/ant_el_step)+1);
        end
        simazel_prev = simazel(i,2);
        continue;
    end

    if(ant_is_moving == false)
        simazel(i,2) = ant_azel(i,2);

        future_el = elev(min(i+floor((ant_t_el/2)*f_vz),length(elev)));
        motor_el = ant_el_step*(round(elev(i)/ant_el_step)) +
ant_v_el*(ant_t_el/2);
        if(future_el > motor_el)
            ant_is_moving = true;
            if i < 100
                %sprintf("%g > %g => Moving UP %i", motor_el, future_el, i)
            end
        end
    end
    end %if

end

figure(1)
subplot(312), stem(t,simazel(:,2)), hold on, plot(t,elev), hold off
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sim", "Ideal", "Konst.")
subplot(313), stem(t,simazel(:,2)), hold on, plot(t,elev), hold off
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sim", "Ideal", "Konst.")
```

```
f_vz = 1;   % sampling frequency


% parameters of satellite
v_el = 0.167;       % angular rate of elevation of the satellite
t_max = 180/v_el;   % time it takes for the satellite to end its journey
azim = 0;      % initial azimuth of the satellite

t = 0:1/f_vz:t_max; % time vector
tlen = length(t);
elev = horzcat(v_el*t(1:tlen/2), -v_el*t(tlen/2+1:end) + 180);      % vector of
elevations of the satellite in time

figure(1)
subplot(311), plot(t,elev)
title("Az 0-360°, El 0-180°")
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sat")

% capabilities of the tracking system
ant_az_step = 1;      % min. angle step in deg - azimuth
ant_el_step = 1;      % min. angle step in deg - elevation
ant_v_az    = 4.5;    % max. angular rate of azimuth of the antenna
ant_v_el    = 4.5;    % max. angular rate of elevation of the antenna

ant_t_az    = ant_az_step/ant_v_az;     % time it takes for the antenna to change
its azimuth direction
ant_t_el    = ant_el_step/ant_v_el;     % time it takes for the antenna to change
its elevation direction

ant_azel = zeros(length(t),2);  % instructions for the tracker motors
ant_azel(1,:) = [azim,0];       % set first position
ant_azel(:,1) = ant_az_step*round(azim/ant_az_step)*ones(length(t),1); % set all
azims to azim

simazel = zeros(length(t),2);   % only for vizualizing
simazel(1,:) = [azim,0];       % set first position
simazel(:,1) = ant_az_step*round(azim/ant_az_step)*ones(length(t),1); % set all
azims to azim
ant_is_moving = false;
for i = 2:length(t)
    % current position of the sattelite - [azim,elev(t)]
    %ant_azel(i,2) = ant_el_step*round(elev(i)/ant_el_step); % simple version

    ant_azel(i,2) = ant_el_step*(round(elev(i)/ant_el_step));
    if(ant_is_moving == true)
        simazel(i,2) = simazel(i-1,2) + ant_v_el/f_vz;
        %ant_azel(i,2) = ant_el_step*(round(elev(i)/ant_el_step)+1);
        if(simazel(i,2) > ant_el_step*(floor(elev(i)/ant_el_step)+1))
            ant_is_moving = false;
            simazel(i,2) = ant_el_step*(floor(elev(i)/ant_el_step)+1);
        end
        continue;
    end

    if(ant_is_moving == false)
        simazel(i,2) = ant_azel(i,2);

        future_el = elev(min(i+floor((ant_t_el/2)*f_vz),length(elev)));
        motor_el = ant_el_step*(round(elev(i)/ant_el_step)) +
ant_v_el*(ant_t_el/2);
        if(future_el > motor_el)
            ant_is_moving = true;
            if i < 100
                sprintf("%g > %g => Moving %i", motor_el, future_el, i)
            end
        end
    end

end

figure(1)
subplot(312), stem(t,simazel(:,2)), hold on, plot(t,elev), hold off
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sim", "Ideal", "Konst.")
subplot(313), stem(t,ant_azel(:,2)), hold on, plot(t,elev), hold off
xlabel("t [s]")
ylabel("Elevation [°]")
legend("Sim", "Ideal", "Konst.")
```