```matlab
function ecoord = latlong(location)
% converts a position specified in ECEF coordinates
% into WGS-84 latitude-longitude-altitude coordinates
%
% input: 'location' vector which contains a position specified by
%       ECEF coordinates (meters)
%                           [ ECEFx ECEFy ECEFz ]
%
% output: 'ecoord' vector which contains the same position specified
%       by WGS-84 latitude (degrees), longitude (degrees), and
%       altitude (meters)
%                           [ latitude longitude altitude ]
%
% define physical constants
    % constant;
% get ECEF location to be converted to latitude-longitude-altitude
% coordinates
    degrad = pi/180.0;
    muearth = 398600.5e9;              % meters^3/second^2
    AA = 6378137.00000;                % meters
    BB = 6356752.31425;                % meters
    esquare=(AA^2 - BB^2) / AA^2;
    ECEFx = location(:,1);
    ECEFy = location(:,2);
    ECEFz = location(:,3);
% compute the longitude which is an exact calculation
    long = atan2(ECEFy , ECEFx);    % radians
% compute the latitude using iteration
    p = sqrt(ECEFx.^2 + ECEFy.^2);
    % compute approximate latitude
    lat0 = atan((ECEFz ./ p) ./ (1 - esquare));
    stop = 0;
    while (stop == 0)
        N0 = AA^2 ./ (sqrt(AA^2 * (cos(lat0)).^2 + ...
             BB^2 .* (sin(lat0)).^2));
        altitude = (p ./ cos(lat0)) - N0;       % meters
        % calculate improved latitude
        term = (1 - esquare * (N0 ./ (N0 + altitude))).^(-1);
        lat = atan(ECEFz ./ p .* term);         % radians
        % check if result is close enough,
        if (abs(lat - lat0) <  1.0e-12)
            stop = 1;
        end
        lat0 = lat;
    end
% convert the latitude and longitude to degrees
    latitude = lat ./ degrad;       % degrees
    longitude = long ./ degrad;     % degrees
% return location in latitude-longitude-altitude coordinates
    ecoord = [ latitude longitude altitude ];
    return;
```