```
function [v, gamma] = cart2polar(vec_r,vec_t, opt)
    % vec_r and vec_t must be the same size
    % opt 1-4 = quad 1-4 respectively

    arguments
        vec_r
        vec_t
        opt.Quad (1,1) = 0;
    end

    % Makes sure that opt is possible
    if ~ismember(opt.Quad,[0 1 2 3 4])
        opt.Quad = 0;
    end

    % Magnitude of vector, vectorized to handle 3D arrays
    v = single((vec_r.^2+vec_t.^2).^(1/2));

    switch opt.Quad
        case 0
            % Returns sine and cosine values
            % Out puts as arrays

            % Sine
            theta_s1 = single(asin(vec_r./v));
            theta_s2 = single(pi-asin(vec_r./v));

            % Cosine
            theta_c1 = single(acos(vec_t./v));
            theta_c2 = single(-acos(vec_t./v));

            % Predefines array size
            gamma = single(zeros(size(vec_r)));

            % Crates logical matrixes that meet the different critera of the quads
            % Assigns gammas based on the logical matrixes
            % Clears unused array
            % Repeat

            % Quad 1
            loc1 = abs(theta_s1(:,:,:) - theta_c1(:,:,:)) < 1.e-5;
            gamma(loc1) = theta_s1(loc1);
            clear loc1

            % Quad 2
            loc2 = abs(theta_s2(:,:,:) - theta_c1(:,:,:)) < 1.e-5;
            gamma(loc2) = theta_s2(loc2);
            clear loc2

            % Quad 3
            loc3 = abs(theta_s2(:,:,:) - theta_c2(:,:,:)) < 1.e-5;
            gamma(loc3) = theta_s2(loc3);
            clear loc3

            % Quad 4
```

```
            loc4 = abs(theta_s1(:,:,:) + theta_c1(:,:,:)) < 1.e-5;
            gamma(loc4) = theta_c2(loc4);
            clear loc4

        case 1
            % Finds quad 1 only
            theta_s1 = single(asin(vec_r./v));
            gamma = theta_s1;

        case 2
            % Finds quad 2 only
            theta_s2 = single(pi-asin(vec_r./v));
            gamma = theta_s2;

        case 3
            % Finds quad 3 only
            theta_s2 = single(pi-asin(vec_r./v));
            gamma = theta_s2;

        case 4
            % Finds quad 4 only
            theta_c2 = single(-acos(vec_t./v));
            gamma = theta_c2;

    end
    clear theta_s1 theta_s2 theta_c1 theta_c2 vec_r vec_t
end
```