Electromagnetic Waves

ECE242

(Project)

Supervised by:

Dr. Hussein Kotb

Eng. Mona Kamel

Made by: Team (1)

Ahmed Sherif Mohamed 23P0414

Ahmed Belal Abdelrahman 23P0007

Amr Ahmed Zaki 23P0011

Contents

Introduction

This project simulates the propagation and focusing of a Gaussian beam using the Angular Spectrum Method (ASM) in MATLAB. The simulation models how the beam propagates in free space, interacts with a thin lens, and focuses at the expected focal point. The goal is to understand and visualize how diffraction and phase transformations affect Gaussian beam profiles during propagation.

A Gaussian beam is initially defined at z=0 with a circularly symmetric intensity distribution, based on a user-defined beam waist w0w_0w0. The electric field distribution is represented as:

$$U(x, y, z = 0) = A_0 \cdot \exp\left(-\frac{x^2 + y^2}{w_0^2}\right)$$

The field is then transformed into the spatial frequency domain using a 2D Fourier Transform. Beam propagation in free space is simulated by applying a phase shift in the frequency domain, based on the propagation distance z, and the inverse Fourier Transform is used to recover the spatial domain field after propagation.

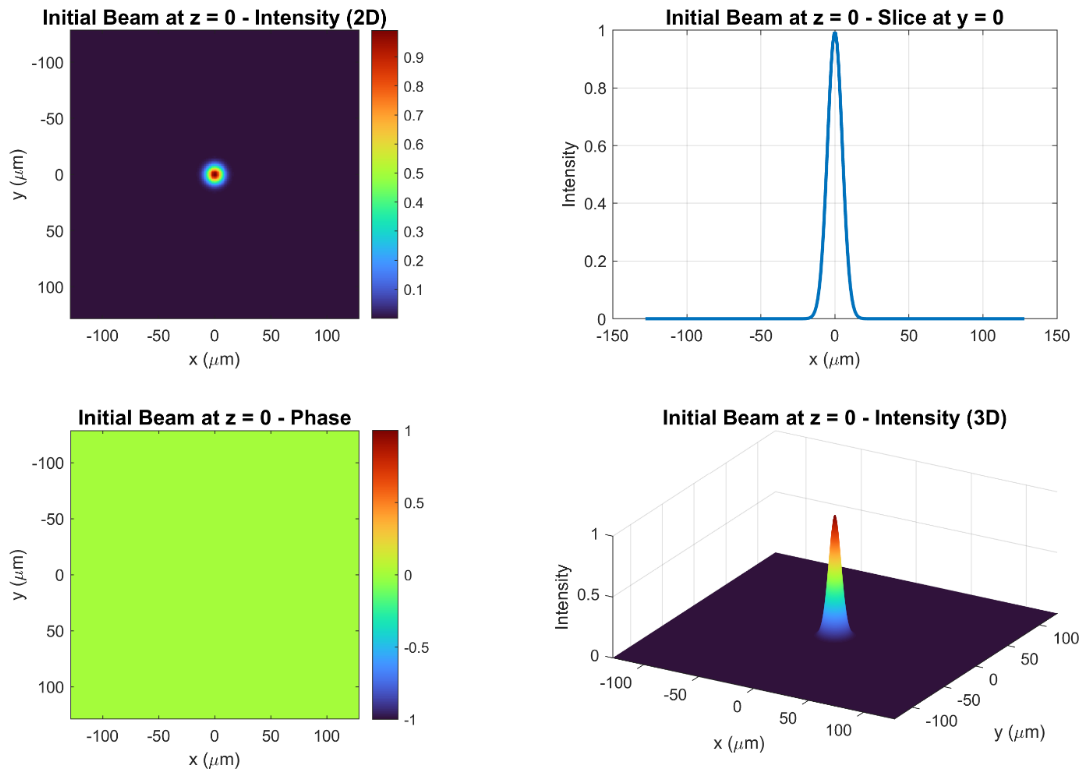
The beam is propagated to the following key positions before the lens:

- z=0 — the beam's initial profile
- z=z0 — one Rayleigh range (where diffraction becomes noticeable)
- z=2z0 — where the lens will be applied

Figures

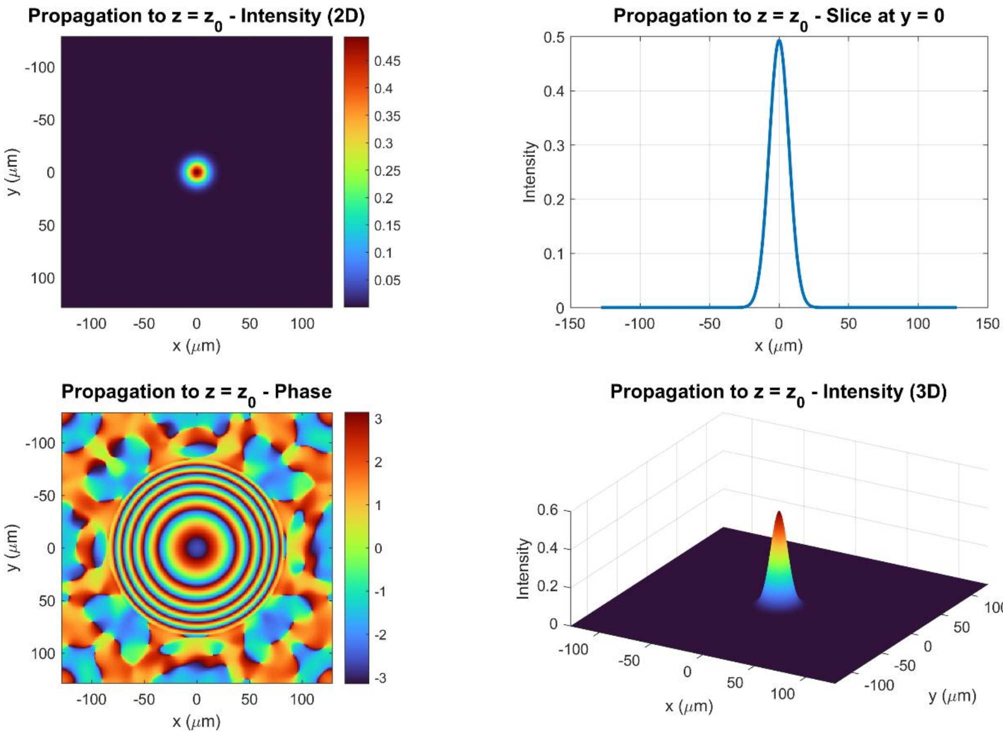
Initial Beam at z=0:

The beam has a perfect circular Gaussian shape cantered at the origin. This is expected because the field was initialized with a Gaussian function. The intensity is highest at the centre and gradually decreases outward, forming a smooth bell-shaped profile.



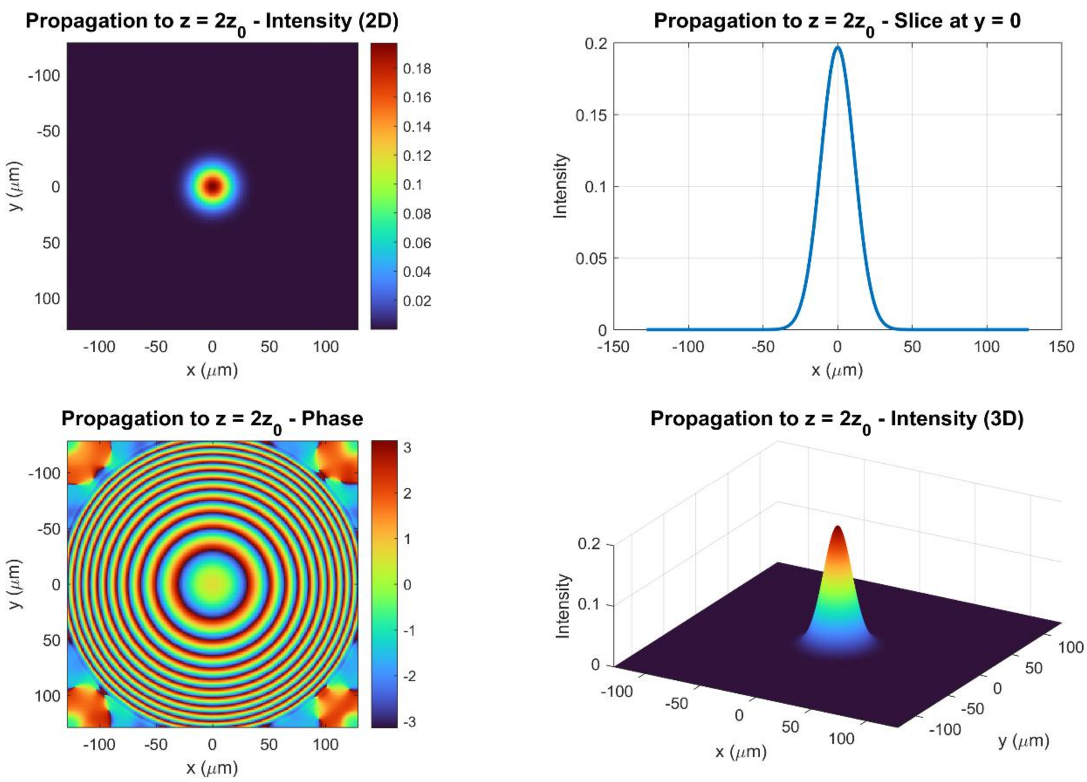
Beam at z=z0:

At this distance, the beam has started to expand due to natural wave spreading. The central peak intensity has decreased, and the beam width has slightly increased. This is expected because waves naturally spread out as they travel in space.



Beam at z=2z0:

The wave has spread even more, showing lower intensity at the center and a wider profile. This behavior is predictable since the beam keeps expanding as it moves farther away from the source in free space.



At z=2z0, the beam passes through a thin lens, which introduces a quadratic phase shift defined by the transmission function:

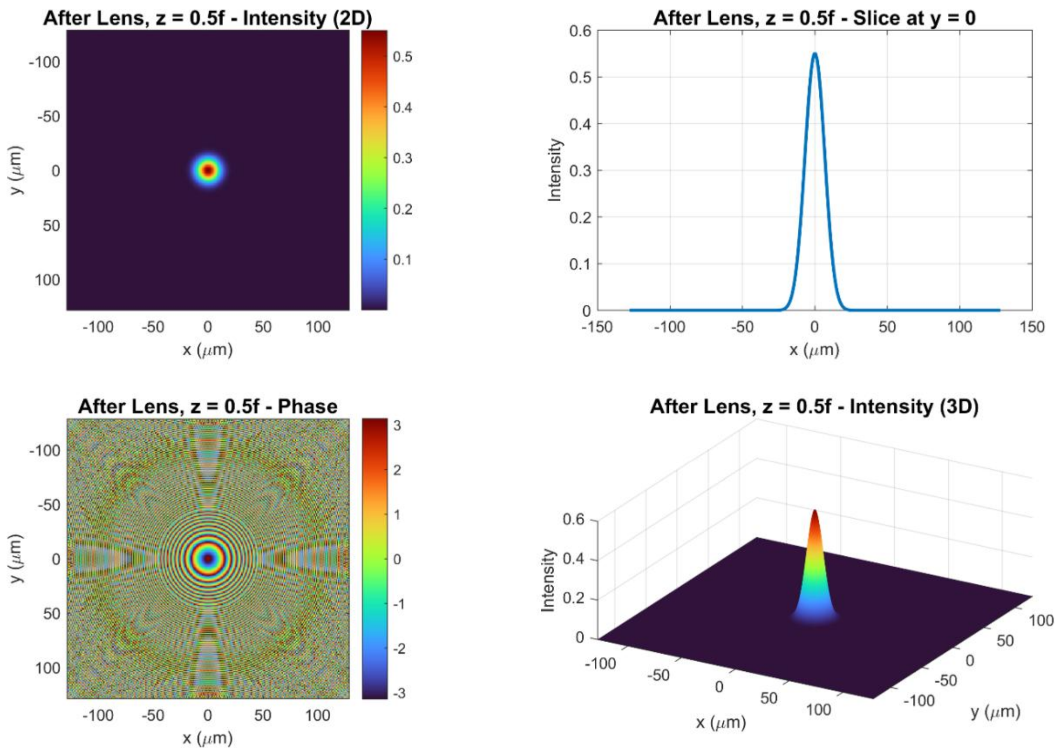
$$T(x, y) = \exp\left(\frac{jk}{2f}(x^2 + y^2)\right)$$

Here, f=0.5z0 is the lens's focal length. After applying the lens phase transformation, the beam is propagated further to three key locations:

- z=0.5f — approaching focus
- z=f — focal point (maximum beam concentration)
- z=2f — beam diverges again after focus

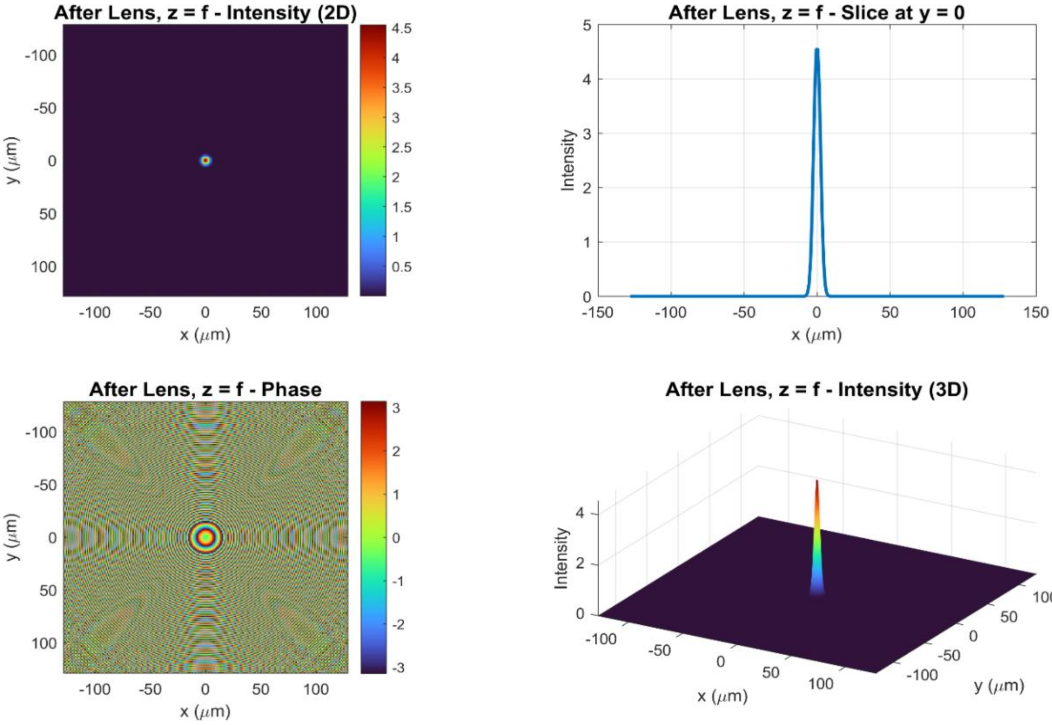
After Lens at z=0.5f:

After passing through the lens, the beam begins to concentrate again. At this distance, the wave is still converging — the beam width is shrinking and intensity is rising. This is expected because the lens is designed to focus the beam on a specific point.



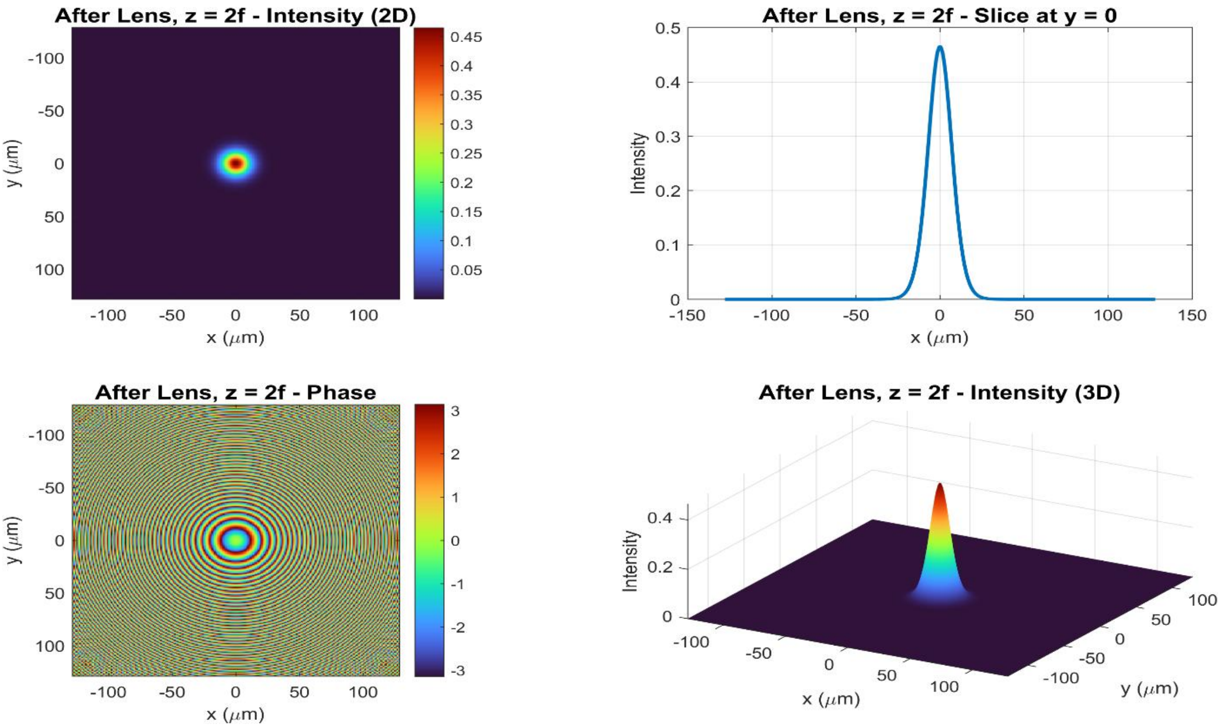
After Lens at z=f:

At the focal point, the beam becomes most concentrated. The intensity is highest and the beam width is at its smallest. This is exactly what we expect from a focused beam — the lens has redirected the spreading wave back into a narrow region.



After Lens at z=2f:

Beyond the focus point, the beam starts to expand again. The intensity at the center decreases and the beam gets wider. This symmetric spreading is expected because after being focused, the wave continues to travel and spread in space just like it did before the lens.



In each plot, the simulation shows:

- A 2D intensity map
- A 1D cross-section of the intensity at y=0
- The phase distribution
- A 3D surface of intensity

The MATLAB simulation uses the following functions:

- fft2, ifft2 for Fourier transforms
- meshgrid, imagesc, surf for spatial grids and visualization
- exportgraphics for saving figures
- VideoWriter for generating an animation

An animation is also created to visualize the beam's evolution throughout its propagation path, both before and after the lens: Drive Link

This simulation demonstrates key principles of optical wave propagation, diffraction, and focusing, making it a valuable tool for understanding Gaussian optics.

Code

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx * | +

```
%% Gaussian Beam Propagation and Focusing - Team 1 (Final Version)
% Author: Team 1 - Spring 2025
% Description: This script simulates the propagation of a Gaussian beam in free space
%              and through a lens, demonstrating beam focusing behavior
% Includes: Beam propagation calculation, lens focusing, phase plots, animation, exports

%% Main Script Structure
% 1. Define beam and simulation parameters
% 2. Initialize spatial grid and initial beam profile
% 3. Simulate free space propagation
% 4. Apply lens and simulate focusing
% 5. Create comprehensive animation
% 6. Utility functions for propagation and visualization

%% 1. BEAM AND SIMULATION PARAMETERS
% Physical parameters
lambda0 = 2e-6;        % Free-space wavelength [m]
w0 = 10e-6;            % Initial beam waist [m]
epsilon_r = 1;         % Relative permittivity of medium (air)
lambda = lambda0 / sqrt(epsilon_r);    % Wavelength in medium [m]
k = 2 * pi / lambda;    % Wavenumber [rad/m]
z0 = pi * w0^2 / lambda;  % Rayleigh range [m] - characteristic distance for beam expansion
f = 0.5 * z0;          % Lens focal length [m]

% Simulation grid parameters
dx = 1e-6;             % Spatial step size [m]
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx * | +

```
L = 256e-6;            % Total grid size [m]
N = round(L / dx);      % Number of grid points
x = linspace(-L/2, L/2, N);  % x-axis grid [m]
y = x;                 % y-axis grid [m] (symmetric)
[X, Y] = meshgrid(x, y);  % 2D grid coordinates [m]

%% 2. INITIAL BEAM PROFILE
% Calculate initial Gaussian beam at z = 0
rho_sq = X.^2 + Y.^2;  % Squared radial distance from beam center [m²]
U0 = exp(-rho_sq / w0^2);  % Field amplitude (Gaussian profile)

% Plot and export initial beam profile
plot_results(X, Y, U0, 0, 'Initial Beam at z = 0');
exportgraphics(gcf, 'fig1_initial.png', 'Resolution', 300);

%% 3. FREE SPACE PROPAGATION
% Propagate to z = z0 (one Rayleigh range)
z = z0;
[Uz_z0, ~] = propagate_beam(U0, k, z, dx, N);
plot_results(X, Y, Uz_z0, z, 'Propagation to z = z_0');
exportgraphics(gcf, 'fig2a_z0.png', 'Resolution', 300);

% Propagate to z = 2z0 (two Rayleigh ranges, where lens will be placed)
z = 2 * z0;
[Uz_2z0, ~] = propagate_beam(U0, k, z, dx, N);
plot_results(X, Y, Uz_2z0, z, 'Propagation to z = 2z_0');
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
[Uz_2z0, ~] = propagate_beam(U0, k, z, dx, N);
plot_results(X, Y, Uz_2z0, z, 'Propagation to z = 2z_0');
exportgraphics(gcf, 'fig2b_2z0.png', 'Resolution', 300);

%% 4. LENS APPLICATION AND FOCUSING
% Apply lens phase mask at z = 2z0
T = exp(1i * (k / (2*f)) * (X.^2 + Y.^2));  % Thin lens transmission function
U_after_lens = Uz_2z0 .* T;  % Apply lens to beam

% Define propagation distances after lens
distances = [0.5*f, f, 2*f];  % [m] - half focal length, focal length, twice focal length
titles = {'After Lens, z = 0.5f', 'After Lens, z = f', 'After Lens, z = 2f'};
filenames = {'fig3_0.5f.png', 'fig4_f.png', 'fig5_2f.png'};

% Propagate beam after lens to each distance and visualize
for i = 1:3
    z = distances(i);
    [U_prop, ~] = propagate_beam(U_after_lens, k, z, dx, N);
    plot_results(X, Y, U_prop, z, titles{i});
    exportgraphics(gcf, filenames{i}, 'Resolution', 300);
end

%% 5. ANIMATION CREATION
% Create comprehensive animation showing beam propagation and focusing
create_beam_animation(U0, U_after_lens, k, dx, N, X, Y, x, y, z0, f);

```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
%% =================== UTILITY FUNCTIONS ====================

%PROPAGATE_BEAM Propagates a complex field using the Angular Spectrum Method
%   Uses the Angular Spectrum Method to propagate an input complex field
%   U_input to a distance z, given wavenumber k and spatial parameters
%
%   Parameters:
%       U_input - Input complex field amplitude
%       k       - Wavenumber [rad/m]
%       z       - Propagation distance [m]
%       dx      - Spatial step size [m]
%       N       - Number of grid points
%
%   Returns:
%       Uz      - Complex field amplitude at distance z
%       Iz      - Intensity at distance z

function [Uz, Iz] = propagate_beam(U_input, k, z, dx, N)

    % Forward FFT with proper centering for angular spectrum calculation
    U_k = fftshift(fft2(ifftshift(U_input)));

    % Frequency space grid
    kx = linspace(-pi/dx, pi/dx - 2*pi/(N*dx), N);
    [Kx, Ky] = meshgrid(kx, kx);
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
    kx = linspace(-pi/dx, pi/dx - 2*pi/(N*dx), N);
    [Kx, Ky] = meshgrid(kx, kx);

    % Transfer function for propagation in k-space
    kz = sqrt(k^2 - Kx.^2 - Ky.^2);     % z-component of wavevector
    H = exp(-1i * real(kz) * z);       % Propagation phase factor (ignore evanescent waves)

    % Apply transfer function and compute inverse FFT to get spatial field
    Uz_k = U_k .* H;
    Uz = fftshift(ifft2(ifftshift(Uz_k)));
    Iz = abs(Uz).^2;  % Calculate intensity
end

%PLOT_RESULTS Creates a 4-panel figure visualizing beam properties
%   Generates plots showing intensity (2D and 3D), phase, and a 1D intensity slice
%
%   Parameters:
%       X, Y      - Spatial grids [m]
%       U         - Complex field amplitude
%       z         - Propagation distance [m]
%       title_str - Title for the plots

function plot_results(X, Y, U, ~, title_str)

    figure('Name', title_str, 'NumberTitle', 'off', 'Color', 'w', 'Position', [200, 40, 1100, 700]);

```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab

    % Panel 1: 2D Intensity distribution
    subplot(2,2,1);
    imagesc(X(1,:)*1e6, Y(:,1)*1e6, abs(U).^2);
    axis image; colormap hot; colorbar;
    title([title_str, ' - Intensity (2D)'], 'FontSize', 13);
    xlabel('x (\mum)'); ylabel('y (\mum)');

    % Panel 2: Intensity slice along x-axis (y = 0)
    subplot(2,2,2);
    plot(X(1,:)*1e6, abs(U(round(end/2), :)).^2, 'LineWidth', 2);
    grid on;
    xlabel('x (\mum)'); ylabel('Intensity');
    title([title_str, ' - Slice at y = 0'], 'FontSize', 13);

    % Panel 3: Phase distribution
    subplot(2,2,3);
    imagesc(X(1,:)*1e6, Y(:,1)*1e6, angle(U));
    colormap hsv; colorbar; axis image;
    title([title_str, ' - Phase'], 'FontSize', 13);
    xlabel('x (\mum)'); ylabel('y (\mum)');

    % Panel 4: 3D Intensity surface plot
    subplot(2,2,4);
    surf(X*1e6, Y*1e6, abs(U).^2, 'EdgeColor', 'none');
    view(30, 45); shading interp; colormap turbo;
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
        xlabel('x (\mum)'); ylabel('y (\mum)'); zlabel('Intensity');
        title([title_str, ' - Intensity (3D)'], 'FontSize', 13);
    end

    %CREATE_BEAM_ANIMATION Creates an animation of beam propagation and focusing
    %   Generates a video showing the beam propagation before and after the lens
    %
    %   Parameters:
    %       U0           - Initial beam profile at z=0
    %       U_after_lens - Complex field after lens application
    %       k            - Wavenumber [rad/m]
    %       dx           - Spatial step [m]
    %       N            - Number of grid points
    %       X, Y         - Spatial grids [m]
    %       x, y         - 1D spatial coordinates [m]
    %       z0           - Rayleigh range [m]
    %       f            - Focal length [m]

    function create_beam_animation(U0, U_after_lens, k, dx, N, X, Y, x, y, z0, f)

        % Define the key z positions used in figures
        z_positions = [0, z0, 2*z0];  % Before lens
        z_positions_after_lens = [0.5*f, f, 2*f];  % After lens

        % Create video file
        video = VideoWriter('complete_beam_propagation.avi');
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
        % Create video file
        video = VideoWriter('complete_beam_propagation.avi');
        video.FrameRate = 10;
        open(video);

        % Create figure with 2D and 3D visualizations
        figure('Position', [100, 100, 1000, 600], 'Name', 'Complete Beam Propagation and Focusing');

        % Calculate maximum intensity for consistent color scaling
        max_intensity = 0;
        for i = 1:length(z_positions)
            [Uz_temp, ~] = propagate_beam(U0, k, z_positions(i), dx, N);
            max_intensity = max(max_intensity, max(abs(Uz_temp(:)).^2));
        end

        for i = 1:length(z_positions_after_lens)
            [Uz_temp, ~] = propagate_beam(U_after_lens, k, z_positions_after_lens(i), dx, N);
            max_intensity = max(max_intensity, max(abs(Uz_temp(:)).^2));
        end

        % Define z-steps for smoother animation while matching key positions
        z_before = linspace(0, 2*z0, 20);  % Propagation before lens
        z_after = linspace(0, 2*f, 20);    % Propagation after lens

        % Part 1: Animation before lens
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab

    % Part 1: Animation before lens
    for i = 1:length(z_before)
        z = z_before(i);
        [Uz_anim, ~] = propagate_beam(U0, k, z, dx, N);

        % 2D intensity plot
        subplot(1,2,1);
        imagesc(x*1e6, y*1e6, abs(Uz_anim).^2);
        axis image; colormap hot; colorbar;
        title(['Free Space Propagation: z = ' num2str(z*1e6, '%.1f') ' \mum'], 'FontSize', 14);
        xlabel('x (\mum)'); ylabel('y (\mum)');

        % 3D Surface plot
        subplot(1,2,2);
        surf(X*1e6, Y*1e6, abs(Uz_anim).^2, 'EdgeColor', 'none');
        shading interp; colormap turbo;
        view(40, 30);
        title(['3D Intensity at z = ' num2str(z*1e6, '%.1f') ' \mum'], 'FontSize', 14);
        xlabel('x (\mum)'); ylabel('y (\mum)'); zlabel('Intensity');
        zlim([0, max_intensity*1.1]);

        % Add a text indicator for the stage of propagation
        annotation('textbox', [0.25, 0.01, 0.5, 0.05], 'String', 'Free Space Propagation (Before Lens)', ...
            'HorizontalAlignment', 'center', 'BackgroundColor', 'yellow', 'FontSize', 12, 'FaceAlpha', 0.7);

```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
        drawnow;
        writeVideo(video, getframe(gcf));
    end

    % Show the lens application
    [Uz_at_lens, ~] = propagate_beam(U0, k, 2*z0, dx, N);
    for j = 1:3  % Show the lens application for 3 frames
        % 2D intensity plot
        subplot(1,2,1);
        imagesc(x*1e6, y*1e6, abs(Uz_at_lens).^2);
        axis image; colormap hot; colorbar;
        title('Beam at Lens Position (z = 2z_0)', 'FontSize', 14);
        xlabel('x (\mum)'); ylabel('y (\mum)');

        % 3D Surface plot
        subplot(1,2,2);
        surf(X*1e6, Y*1e6, abs(Uz_at_lens).^2, 'EdgeColor', 'none');
        shading interp; colormap turbo;
        view(40, 30);
        title('3D Intensity at Lens Position', 'FontSize', 14);
        xlabel('x (\mum)'); ylabel('y (\mum)'); zlabel('Intensity');
        zlim([0, max_intensity*1.1]);

        % Text indicator for lens application
        annotation('textbox', [0.25, 0.01, 0.5, 0.05], 'String', 'Lens Application at z = 2z_0', ...
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab
                'HorizontalAlignment', 'center', 'BackgroundColor', 'cyan', 'FontSize', 12, 'FaceAlpha', 0.7);

            drawnow;
            writeVideo(video, getframe(gcf));
        end

        % Part 2: Animation after lens
        for i = 1:length(z_after)
            z = z_after(i);
            [Uz_anim, ~] = propagate_beam(U_after_lens, k, z, dx, N);

            % 2D intensity plot
            subplot(1,2,1);
            imagesc(x*1e6, y*1e6, abs(Uz_anim).^2);
            axis image; colormap hot; colorbar;
            title(['After Lens: z = ' num2str(z*1e3, '%.1f') ' mm'], 'FontSize', 14);
            xlabel('x (\mum)'); ylabel('y (\mum)');

            % 3D Surface plot
            subplot(1,2,2);
            surf(X*1e6, Y*1e6, abs(Uz_anim).^2, 'EdgeColor', 'none');
            shading interp; colormap turbo;
            view(40, 30);
            title(['3D Intensity at z = ' num2str(z*1e3, '%.1f') ' mm After Lens'], 'FontSize', 14);
            xlabel('x (\mum)'); ylabel('y (\mum)'); zlabel('Intensity');
            zlim([0, max_intensity*1.1]);
```

Editor - C:\Users\amrah\OneDrive\Desktop\Waves_Project.m

Waves_Project.m | untitled.mlx *

```matlab

            % 3D Surface plot
            subplot(1,2,2);
            surf(X*1e6, Y*1e6, abs(Uz_anim).^2, 'EdgeColor', 'none');
            shading interp; colormap turbo;
            view(40, 30);
            title(['3D Intensity at z = ' num2str(z*1e3, '%.1f') ' mm After Lens'], 'FontSize', 14);
            xlabel('x (\mum)'); ylabel('y (\mum)'); zlabel('Intensity');
            zlim([0, max_intensity*1.1]);

            % Special indication when reaching focal point
            if abs(z - f) < f/20  % If we're close to the focal point
                annotation('textbox', [0.25, 0.01, 0.5, 0.05], 'String', 'Focal Point! (z = f)', ...
                    'HorizontalAlignment', 'center', 'BackgroundColor', 'red', 'FontSize', 12, 'FaceAlpha', 0.7);
            else
                annotation('textbox', [0.25, 0.01, 0.5, 0.05], 'String', 'Focusing After Lens', ...
                    'HorizontalAlignment', 'center', 'BackgroundColor', 'green', 'FontSize', 12, 'FaceAlpha', 0.7);
            end

            drawnow;
            writeVideo(video, getframe(gcf));
        end
        close(video);
    end
```

References

[1] D. M. Pozar, *Microwave Engineering*, 4th ed. Hoboken, NJ, USA: Wiley, 2011.

[2] F. T. Ulaby and U. Ravaioli, *Fundamentals of Applied Electromagnetics*, 7th ed. Boston, MA, USA: Pearson, 2014.

THANKS!