# Applications of Aliasing

## Rotating Blades

Consider the rotation of a fan. You record the motion with a camera whose frame rate is set to 24 fps. What do you see?

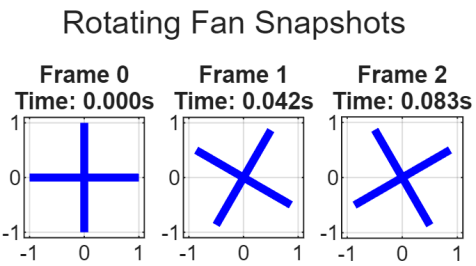
First, we look at a few frames. Change dotshow to 1 to follow first blade.

```
% Wagon Wheel Effect Simulation
fan_hz = 30;    % Real speed of the fan
fps = 24;       % Camera frame rate
n_spokes = 4;   % Number of fan blades/spokes
dotshow = 0;    % Mark fan blades
fan_hz = 28;    % Slow fan

% Calculate perceived motion
rotations_per_frame = fan_hz / fps;
fractional_rot = mod(rotations_per_frame, 1);

if fractional_rot > 0.5
    disp('Prediction: The fan will appear to spin BACKWARD.');
elseif fractional_rot == 0 || fractional_rot == 0.5
    disp('Prediction: The fan will appear STATIONARY or FLICKERING.');
else
    disp('Prediction: The fan will appear to spin FORWARD.');
end
```

```
Prediction: The fan will appear to spin FORWARD.
```

```
% Visualizing 3 consecutive frames
figure('Color', 'w');
t_frame = 1/fps;


for i = 0:2
    subplot(1, 3, i+1);

    % The angle of the first spoke at this frame time
    % Angle = (Time * Speed * 360 degrees)
    current_angle = i * t_frame * fan_hz * 360;

    % Draw the spokes
    for s = 0:n_spokes-1
        angle_rad = deg2rad(current_angle + (s * 360/n_spokes));
        plot([0 cos(angle_rad)], [0 sin(angle_rad)], 'b', 'LineWidth', 3);

        % Add this inside your loop to see the "Real" movement
        plot([0 cos(angle_rad)], [0 sin(angle_rad)], 'b', 'LineWidth', 3);
```

```
        hold on;
        if (s == 0) && (dotshow==1)     % Put a red dot on the tip of the first
spoke
            plot(cos(angle_rad), sin(angle_rad), 'ro', 'MarkerFaceColor', 'r');
        end
    end

    title(sprintf('Frame %d\nTime: %.3fs', i, i*t_frame));
    axis([-1.1 1.1 -1.1 1.1]); axis square; grid on;
    sgtitle("Rotating Fan Snapshots", 'FontSize', 12);
    hold off
end
```




## Wagon Wheel Effect

Next, animate the fan blades. Here we see the **Wagon Wheel Effect**. This is often seen in movies - notably in westerns. In next demo we can ***change the fan frequency*** of rotation.

```
clear
wagon_wheel_demo;
```

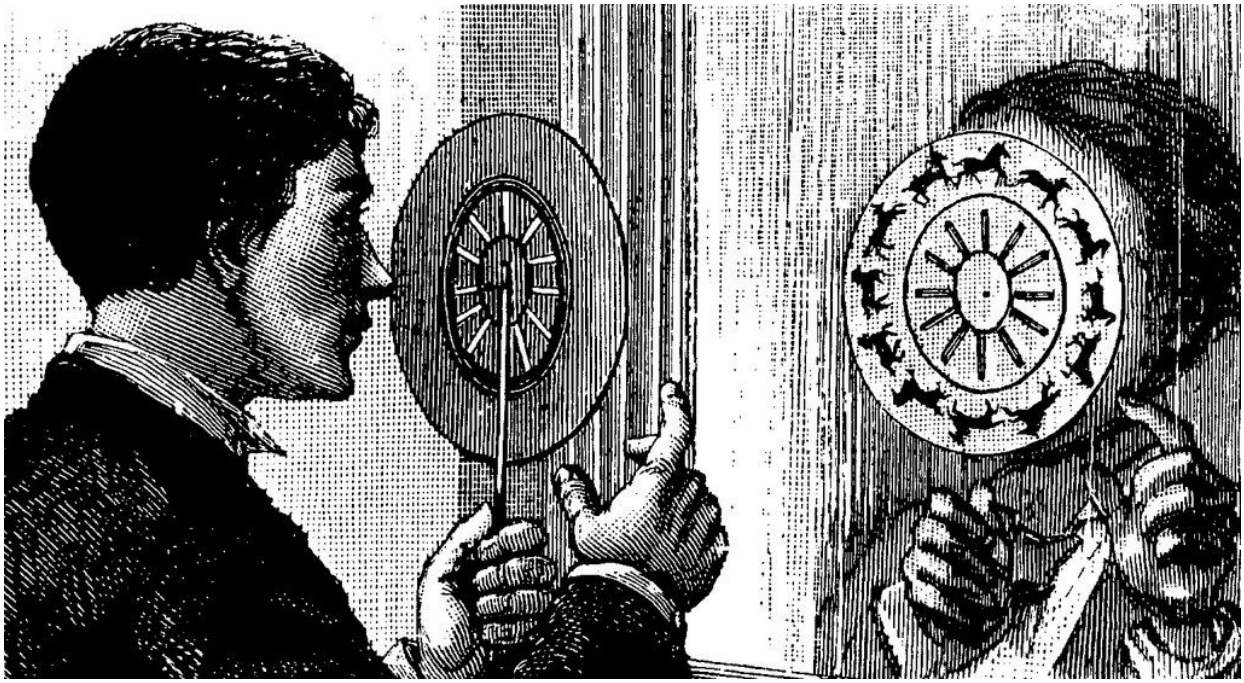
We can also ***change the number of blades***. Note: Instead of fan blades, you can think helicopter blades or spokes on a bicycle or car wheels.

In 1968 there was a movie called ***Bullitt*** starring Steve McQueen in a car chase in San Francisco. Check out the video https://www.youtube.com/watch?v=hq8YD7-Aimw around the 45s position.

```
clear
ultimate_aliasing_demo;
```

## Stroboscopic Effect

This is also referred to as the Stroboscopic effect. The next application shows a Stroboscopic Disk. Wikipedia: "The **stroboscopic disc** was the first tool that showed a moving picture. It is also known as **phénakisticope** or **phenakistoscope**." https://www.mhs.ox.ac.uk/exhibits/fancy-names-and-fun-toys/phenakistiscopes/index.html

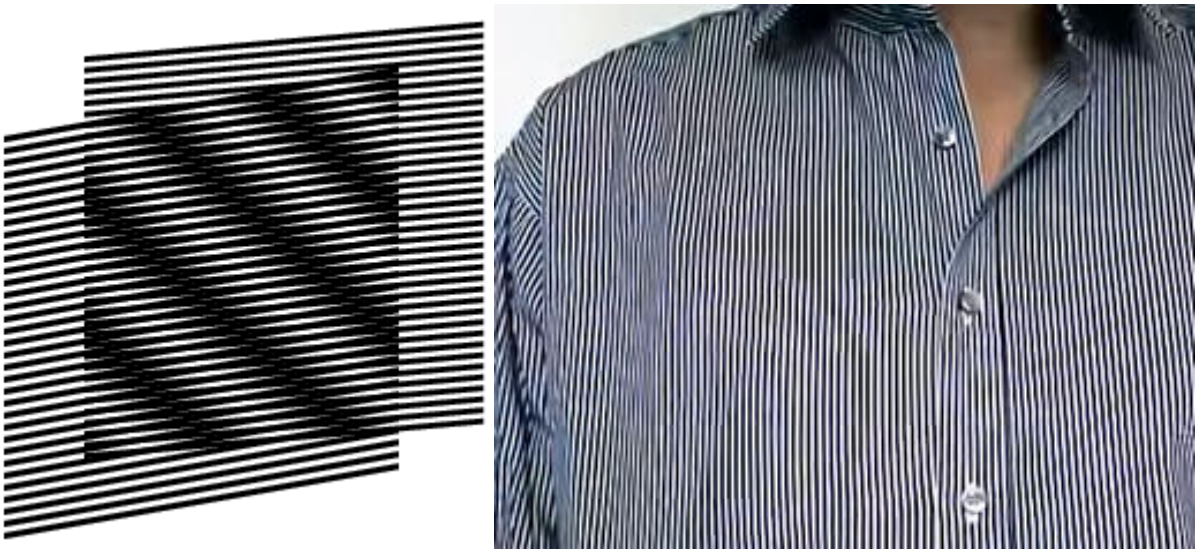


In the next application we have a disk with several rings with black and white sectors of differing sizes. Sampling at 24 fps leads to interesting results generalizing fan rotation with differing numbers of blades.

```
clear
sector_disk_demo;
```

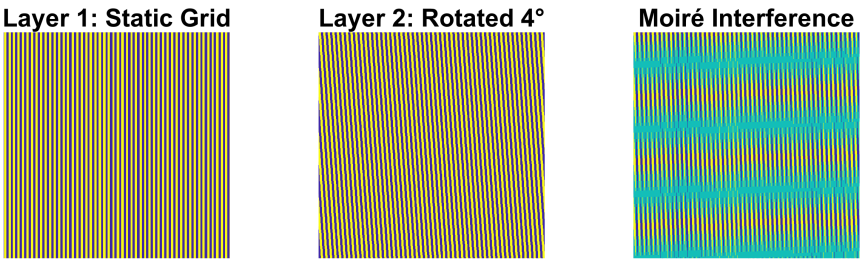
## Moiré Effect

Moiré patterns occurs when **a scene or an object that is being photographed contains repetitive details** (such as lines, dots, etc) that exceed sensor resolution. https://wp.optics.arizona.edu/oscoutreach/moire-patterns/



Run next cell and enlarge the resulting figure. Here we overlay two grids and observe new patterns.

```
% Moiré Effect Classroom Demo
% 1. Setup the spatial domain (a high-resolution canvas)
[x, y] = meshgrid(0:0.01:10, 0:0.01:10);

% 2. Define parameters
f = 5;              % Frequency of the lines (number of stripes)
angle_deg = 4;      % Rotation angle (Try changing this to 1 or 10!)

% 3. Create the first vertical grid (Grid A)
gridA = sin(2 * pi * f * x) > 0;

% 4. Create the second rotated grid (Grid B)
theta = deg2rad(angle_deg);
x_rot = x * cos(theta) - y * sin(theta);
gridB = sin(2 * pi * f * x_rot) > 0;

% 5. Combine grids to show interference (The Moiré Effect)
% We use addition to simulate light passing through two physical layers
moire = gridA + gridB;

% 6. Visualization
figure('Name', 'Moiré Effect Classroom Demo', 'NumberTitle', 'off');
subplot(1,3,1); imshow(gridA); title('Layer 1: Static Grid');
subplot(1,3,2); imshow(gridB); title(['Layer 2: Rotated ', num2str(angle_deg),
'°']);
subplot(1,3,3); imshow(moire, []); title('Moiré Interference');
colormap('Parula'); % Use a high-contrast colormap for visibility
```




## Animation

Next, we show the results of changing the relative angle between the grids.

```
% Moiré Animation for Classroom Demo
[x, y] = meshgrid(0:0.01:10, 0:0.01:10);
f = 4; % Line frequency
grid1 = sin(2 * pi * f * x) > 0;

figure('Color', 'w');
hImg = imshow(grid1, []);
colormap('Parula');
hTitle = title('Moiré Pattern Rotation: 0.0°', 'FontSize', 12);



% Animate from 0.5 to 15 degrees
for angle = [0.5:0.2:15, 15:-0.2:0.5]
    theta = deg2rad(angle);
    x_rot = x * cos(theta) - y * sin(theta);
    grid2 = sin(2 * pi * f * x_rot) > 0;

    % Combine layers
    moire = grid1 + grid2;
```

```
    % Update display
    set(hImg, 'CData', moire);
    hTitle.String = sprintf('Moiré Pattern Rotation: %.1f°', angle);
    drawnow; % Forces MATLAB to update the figure
end
```

### Functions Used

```
function wagon_wheel_demo
    % Parameters
    fps = 24;
    n_spokes = 5;      % <--- You can now change this to 3, 5, 8, etc.!
    t_frame = 1/fps;

    % Create the figure
    fig = figure('Name', 'Wagon Wheel Effect', 'Color', 'w', 'MenuBar', 'none');
    ax = axes('Parent', fig, 'Position', [0.15, 0.3, 0.7, 0.6]);
    axis(ax, [-1.2 1.2 -1.2 1.2]); axis square; grid on;
    hold(ax, 'on');

    % Calculate spacing between spokes dynamically
    spoke_spacing = 360 / n_spokes;

    % Initialize the spokes
    colors = lines(n_spokes);
    hSpokes = gobjects(n_spokes, 1);
    for s = 1:n_spokes
        hSpokes(s) = plot(ax, [0 0], [0 0], 'Color', colors(s,:), 'LineWidth', 4);
    end
    hDot = plot(ax, 0, 0, 'ro', 'MarkerFaceColor', 'r', 'MarkerSize', 8);

    % Slider for Fan Speed
    sld = uicontrol('Parent', fig, 'Style', 'slider', 'Min', 0, 'Max', 60, ...
        'Value', 30, 'Position', [100 60 360 20]);

    hText = uicontrol('Parent', fig, 'Style', 'text', 'Position', [100 40 360 20],
...
        'String', 'Current Speed: 30.0 Hz', 'BackgroundColor', 'w');

    % Main Animation Loop
    frame_count = 0;
    while ishandle(fig)
        fan_hz = sld.Value;
        set(hText, 'String', sprintf('Current Speed: %.1f Hz', fan_hz));

        total_rot = (frame_count * t_frame) * fan_hz * 360;

        for s = 1:n_spokes
            % Dynamic spacing: (s-1) * (360 / n_spokes)
```

```
            angle_rad = deg2rad(total_rot + (s-1) * spoke_spacing);
            set(hSpokes(s), 'XData', [0 cos(angle_rad)], 'YData', [0
sin(angle_rad)]);
            if s == 1
                set(hDot, 'XData', cos(angle_rad), 'YData', sin(angle_rad));
            end
        end

        frame_count = frame_count + 1;
        pause(4*t_frame);
    end
end

function moire_interactive_demo
    % Setup the canvas
    [x, y] = meshgrid(0:0.01:10, 0:0.01:10);
    f = 4; % Base frequency
    grid1 = sin(2 * pi * f * x) > 0;

    % Create the figure window
    fig = figure('Name', 'Interactive Moire Demo', 'Color', 'w', 'MenuBar', 'none');
    ax = axes('Parent', fig, 'Position', [0.1, 0.2, 0.8, 0.7]);

    % Initial display
    hImg = imshow(grid1, 'Parent', ax);
    title(ax, 'Adjust the Slider to Change the Angle');
    colormap(ax, 'parula');

    % Add the Slider
    sld = uicontrol('Parent', fig, 'Style', 'slider', ...
        'Min', 0.1, 'Max', 15, 'Value', 1, ...
        'Position', [150, 50, 300, 20], ...
        'Callback', @updatePlot);

    % Add a label for the slider
    uicontrol('Parent', fig, 'Style', 'text', 'Position', [150, 70, 300, 20], ...
        'String', 'Rotation Angle (Degrees)', 'BackgroundColor', 'w');

    % The function that runs whenever the slider moves
    function updatePlot(src, ~)
        angle = src.Value;
        theta = deg2rad(angle);
        x_rot = x * cos(theta) - y * sin(theta);
        grid2 = sin(2 * pi * f * x_rot) > 0;

        % Update the image data
        set(hImg, 'CData', grid1 + grid2);
        title(ax, sprintf('Rotation Angle: %.2f°', angle));
    end
end
```

```
function ultimate_aliasing_demo
    % Parameters
    fps = 24;
    t_frame = 1/fps;
    isPaused = false; % Initial state

    % Create the figure
    fig = figure('Name', 'Wagon Wheel Effect: Classroom Demo', 'Color', 'w',
'Position', [100 100 600 680]);
    ax = axes('Parent', fig, 'Position', [0.15, 0.45, 0.7, 0.45]);
    axis(ax, [-1.2 1.2 -1.2 1.2]); axis square; grid on;
    hold(ax, 'on');

    % --- UI Controls ---
    % Play/Pause Button
    btnPause = uicontrol('Style', 'pushbutton', 'String', 'PAUSE', ...
        'Position', [225 30 150 40], 'FontSize', 12, 'FontWeight', 'bold', ...
        'Callback', @togglePause);

    % Speed Slider
    uicontrol('Style', 'text', 'Position', [150 200 300 20], 'String', 'Adjust Fan
Speed (Hz)', 'BackgroundColor', 'w');
    sldSpeed = uicontrol('Style', 'slider', 'Min', 0, 'Max', 60, 'Value', 30,
'Position', [150 180 300 20]);
    hSpeedText = uicontrol('Style', 'text', 'Position', [150 160 300 20], 'String',
'Speed: 30.0 Hz', 'BackgroundColor', 'w');

    % Spoke Slider
    uicontrol('Style', 'text', 'Position', [150 130 300 20], 'String', 'Number of
Spokes', 'BackgroundColor', 'w');
    sldSpokes = uicontrol('Style', 'slider', 'Min', 1, 'Max', 12, 'Value', 4,
'SliderStep', [1/11 1/11], 'Position', [150 110 300 20]);
    hSpokeText = uicontrol('Style', 'text', 'Position', [150 90 300 20], 'String',
'Spokes: 4', 'BackgroundColor', 'w');

    % Rim Checkbox
    chkWheel = uicontrol('Style', 'checkbox', 'Position', [150 75 300 20], ...
        'String', 'Add Wheel Rim', 'BackgroundColor', 'w', 'Value', 0);

    % --- Animation Logic ---
    frame_count = 0;
    hSpokes = []; hDot = []; hRim = [];

    % Callback function for the button
    function togglePause(~, ~)
        isPaused = ~isPaused;
        if isPaused
            set(btnPause, 'String', 'RESUME', 'BackgroundColor', [0.7 1 0.7]);
        else
```

```
            set(btnPause, 'String', 'PAUSE', 'BackgroundColor', [1 0.7 0.7]);
        end
    end

    while ishandle(fig)
        if ~isPaused
            % Get values
            fan_hz = sldSpeed.Value;
            n_spokes = round(sldSpokes.Value);
            show_rim = chkWheel.Value;

            set(hSpeedText, 'String', sprintf('Current Speed: %.1f Hz', fan_hz));
            set(hSpokeText, 'String', sprintf('Number of Spokes: %d', n_spokes));

            % Refresh graphics if spoke count changed
            if length(hSpokes) ~= n_spokes
                delete(hSpokes); delete(hDot);
                colors = lines(n_spokes);
                hSpokes = gobjects(n_spokes, 1);
                for s = 1:n_spokes
                    hSpokes(s) = plot(ax, [0 0], [0 0], 'Color', colors(s,:),
'LineWidth', 4);
                end
                hDot = plot(ax, 0, 0, 'ro', 'MarkerFaceColor', 'r', 'MarkerSize',
8);
            end

            % Handle Rim
            if show_rim && isempty(hRim)
                th = linspace(0, 2*pi, 100);
                hRim = plot(ax, cos(th), sin(th), 'k', 'LineWidth', 3);
            elseif ~show_rim && ~isempty(hRim)
                delete(hRim); hRim = [];
            end

            % Update positions
            total_rot = (frame_count * t_frame) * fan_hz * 360;
            spoke_spacing = 360 / n_spokes;

            for s = 1:n_spokes
                angle_rad = deg2rad(total_rot + (s-1) * spoke_spacing);
                set(hSpokes(s), 'XData', [0 cos(angle_rad)], 'YData', [0
sin(angle_rad)]);
                if s == 1
                    set(hDot, 'XData', cos(angle_rad), 'YData', sin(angle_rad));
                end
            end

            frame_count = frame_count + 1;
        end
```

```
        pause(t_frame); % Keep the loop running so the button stays responsive
    end
end

function sector_disk_demo
    fps = 24;
    t_frame = 1/fps;

    fig = figure('Name', 'Multi-Ring Aliasing', 'Color', 'w', 'Position', [100 100
600 600]);
    ax = axes('Parent', fig, 'Position', [0.1 0.2 0.8 0.7]);
    axis(ax, [-1.2 1.2 -1.2 1.2]); axis square; axis off;
    title(ax, 'Stroboscopic Disc', 'FontSize', 12)
    hold(ax, 'on');

    % Setup Sliders
    sldSpeed = uicontrol('Style', 'slider', 'Min', 0, 'Max', 20, 'Value', 2,
'Position', [150 50 300 20]);
    hText = uicontrol('Style', 'text', 'Position', [150 75 300 20], 'String', 'Disk
Speed: 2.0 Hz', 'BackgroundColor', 'w', 'FontSize', 12);

    % Define 3 rings with different sector counts
    ring_sectors = [4, 10, 30]; % Inner, Middle, Outer
    ring_radii = [0.4, 0.7, 1.0];

    hPatches = [];
    frame_count = 0;

    while ishandle(fig)
        fan_hz = sldSpeed.Value;
        set(hText, 'String', sprintf('Disk Speed: %.1f Hz', fan_hz));
        delete(hPatches); hPatches = [];

        total_rot = (frame_count * t_frame) * fan_hz * 360;

        for r = 1:3
            N = ring_sectors(r);
            R_outer = ring_radii(r);
            R_inner = R_outer - 0.25;

            for s = 1:N
                % Only draw every other sector (Black/White)
                start_ang = total_rot + (s-1)*(360/N);
                end_ang = total_rot + s*(360/N);

                if mod(s, 2) == 0
                    % Create a "wedge" for the black sectors
                    theta = deg2rad(linspace(start_ang, end_ang, 20));
                    x_p = [R_inner*cos(theta), flip(R_outer*cos(theta))];
                    y_p = [R_inner*sin(theta), flip(R_outer*sin(theta))];
```

```
                    hPatches(end+1) = patch(x_p, y_p, 'k', 'EdgeColor', 'none');
                end
            end
        end
        frame_count = frame_count + 1;
        pause(t_frame);
    end
end
```