Nicholas Pittman

CS 443

Program 1 Report

14 February 2023

Table of Contents

Figure 1: Unmodified 24-bit Image

Figure 2: Image quantized to bit depth of [3, 3, 2]

Figure 3: Image quantized to bit depth of [4, 2, 2]

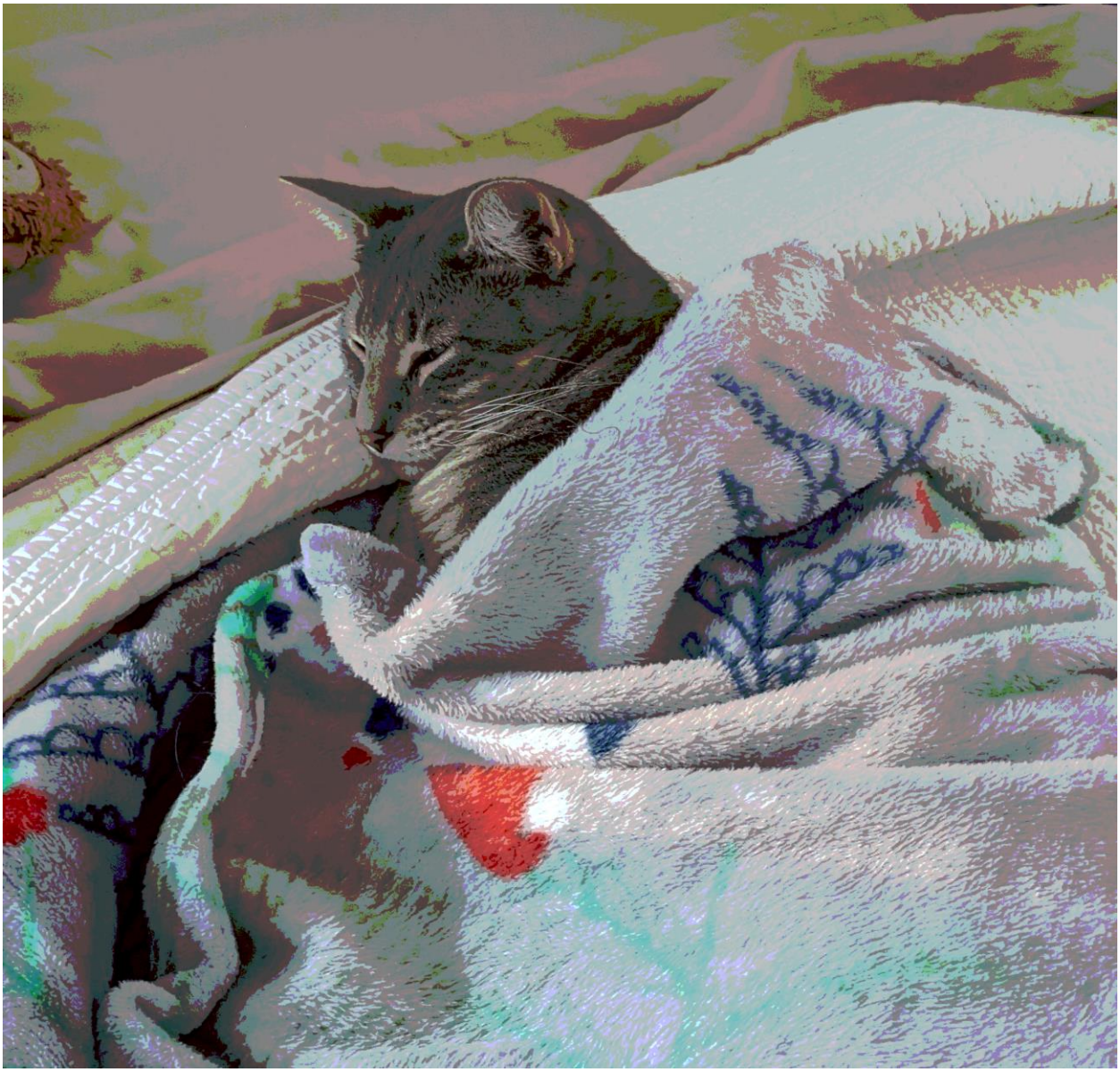


Figure 4: Image quantized to bit depth of [2, 2, 2]

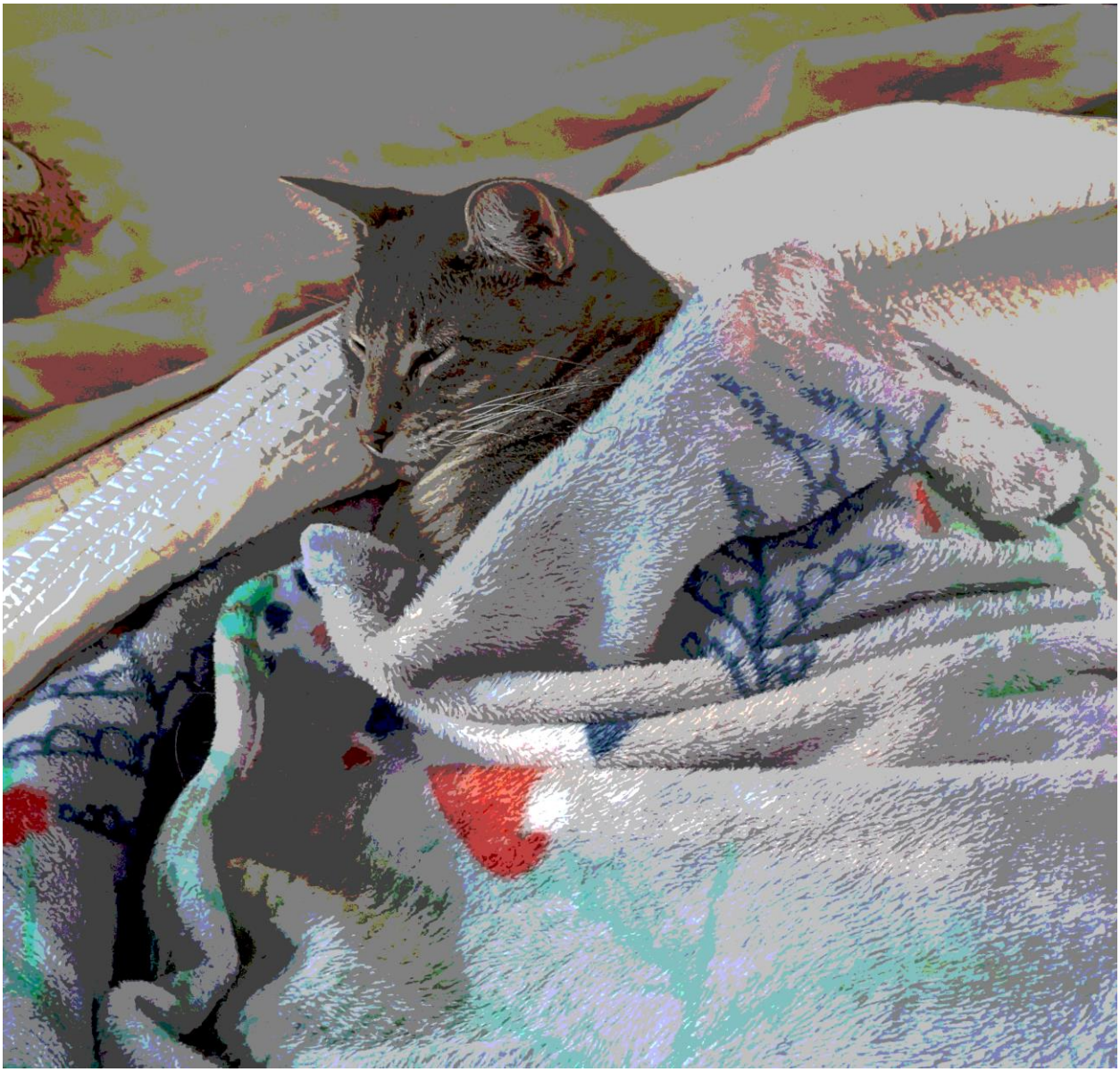
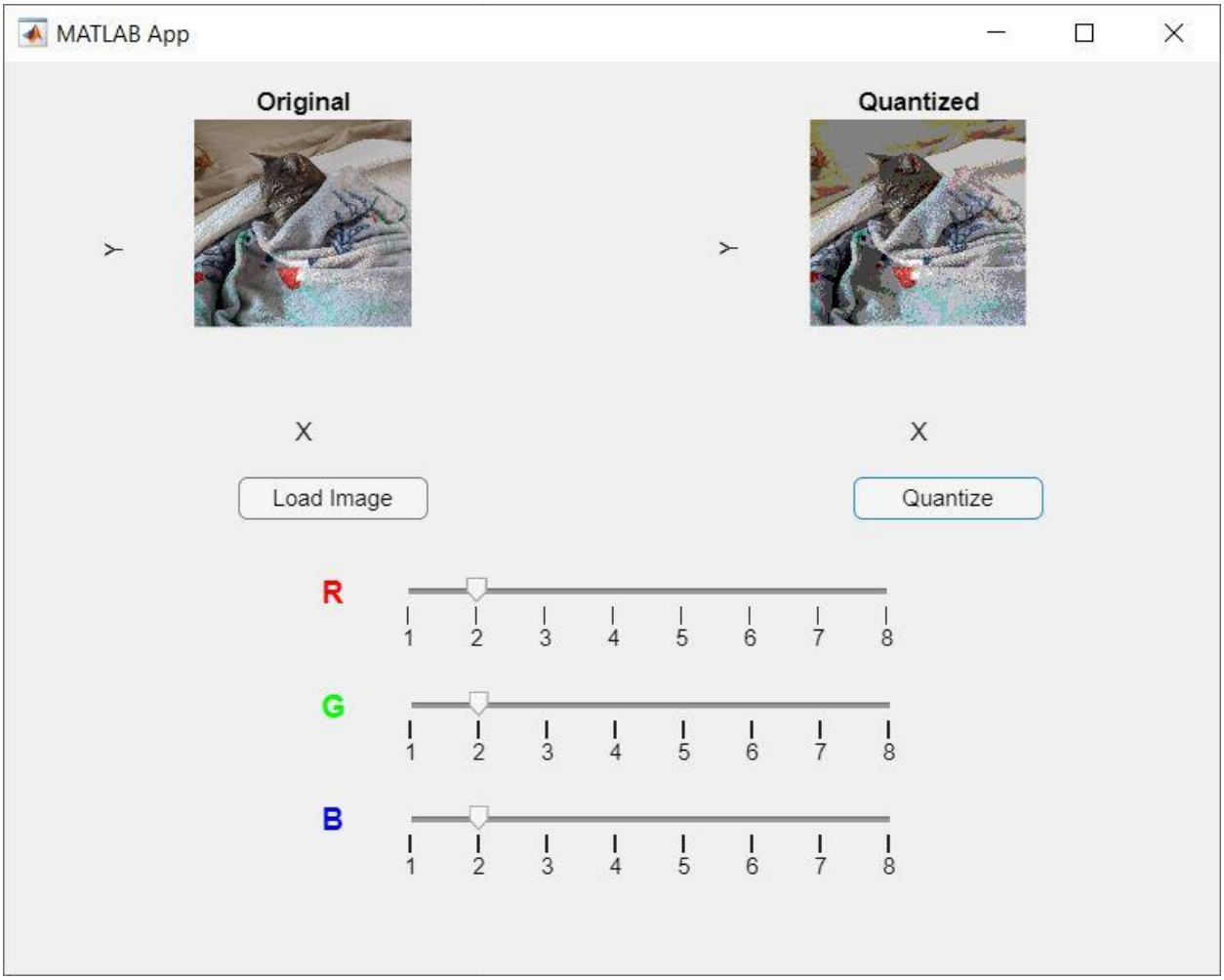


Figure 5: Screenshot of app in action



```
% This function reduces a 24-bit RGB image to the specified bit depth.

% Inputs:
% - image_filename: The path to the source image (must be PNG format)
% - rgb_bit_depth: A 1x3 array specicfying the bit depth of R, G, and B

% Outputs:
% - output_image: The array representing the reduced image.
% This function also writes the output image to the disk in PNG format.
function output_image = uniformQuant(image_filename, rgb_bit_depth)

    % Read in the image
    image = imread(image_filename, 'png');

    [cols, rows] = size(image, 1:2);

    % Calculate the size of the range of colors in each R, G, and B
    % channel that will be reduced to a single color for each bit depth
    partition_size = 256 ./ 2.^uint8(rgb_bit_depth);

    fprintf("Quantizing...\n")
    tic();

    % Iterate over each pixel in the image
    for c = 1:cols
        for r = 1:rows
            % Transform the RGB data into a 1x3 array so it's easier to
            % work with
            rgb = reshape(image(c,r,:),  [1, 3]);

            % Reduce the pixel's color.
            % Divide by the partition size, truncate, then re-multiply by
            % that same partition size
            rgb = floor(rgb ./ partition_size) .* partition_size;

            % Transform the data back to its original configuration
            image(c,r,:) = reshape(rgb, [1, 1, 3]);
        end
    end
    fprintf("Finished after %.2f seconds\n", toc())

    % Write out the image to disk
    output_filename = sprintf("%s_%d_%d_%d.png", image_filename,
 rgb_bit_depth);
    imwrite(image, output_filename)

    % Return
    output_image = image;
end
```

Published with MATLAB® R2022a

```
classdef uniformQuantGUI < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure        matlab.ui.Figure
        BSlider         matlab.ui.control.Slider
        Label_3         matlab.ui.control.Label
        GSlider         matlab.ui.control.Slider
        Label_2         matlab.ui.control.Label
        Quantize        matlab.ui.control.Button
        Load_Image      matlab.ui.control.Button
        BLabel          matlab.ui.control.Label
        GLabel          matlab.ui.control.Label
        RLabel          matlab.ui.control.Label
        RSlider         matlab.ui.control.Slider
        Label           matlab.ui.control.Label
        Quantized_Axes  matlab.ui.control.UIAxes
        Original_Axes   matlab.ui.control.UIAxes
    end


    properties (Access = private)
        input_image
        input_filename
    end


    % Callbacks that handle component events
    methods (Access = private)

        % Value changed function: RSlider
        function RSliderValueChanged(app, event)
            % Snap slider to integers
            app.RSlider.Value = round(app.RSlider.Value);

        end

        % Value changed function: GSlider
        function GSliderValueChanged(app, event)
            % Snap slider to integers
            app.GSlider.Value = round(app.GSlider.Value);
        end

        % Value changed function: BSlider
        function BSliderValueChanged(app, event)
            % Snap slider to integers
            app.BSlider.Value = round(app.BSlider.Value);
        end

        % Button pushed function: Load_Image
        function Load_ImageButtonPushed(app, event)
            % Allow user to select file and read it in
```

```
            app.input_filename = uigetfile('.png');
            focus(app.UIFigure);
            app.input_image = imread(app.input_filename, 'png');

            % Display the image
            imshow(app.input_image, 'Parent', app.Original_Axes);

            % Enable quantize button
            app.Quantize.Enable = true;

        end

        % Button pushed function: Quantize
        function QuantizeButtonPushed(app, event)
            % Get slider values
            RGB = [app.RSlider.Value, app.GSlider.Value, app.BSlider.Value];

            app.Quantize.Text = "Quantizing...";
            app.Quantize.Enable = false;
            drawnow

            % Quantize the image
            output_image = uniformQuant(app.input_filename, RGB);

            app.Quantize.Text = "Quantize";
            app.Quantize.Enable = true;
            drawnow

            % Display the result
            imshow(output_image, 'Parent', app.Quantized_Axes);
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';

            % Create Original_Axes
            app.Original_Axes = uiaxes(app.UIFigure);
            title(app.Original_Axes, 'Original')
            xlabel(app.Original_Axes, 'X')
            ylabel(app.Original_Axes, 'Y')
            zlabel(app.Original_Axes, 'Z')
            app.Original_Axes.Position = [30 278 256 192];

            % Create Quantized_Axes
            app.Quantized_Axes = uiaxes(app.UIFigure);
```

```
            title(app.Quantized_Axes, 'Quantized')
            xlabel(app.Quantized_Axes, 'X')
            ylabel(app.Quantized_Axes, 'Y')
            zlabel(app.Quantized_Axes, 'Z')
            app.Quantized_Axes.Position = [354 278 256 192];

            % Create Label
            app.Label = uilabel(app.UIFigure);
            app.Label.HorizontalAlignment = 'right';
            app.Label.Position = [168 194 25 22];
            app.Label.Text = '';

            % Create RSlider
            app.RSlider = uislider(app.UIFigure);
            app.RSlider.Limits = [1 8];
            app.RSlider.MajorTicks = [1 2 3 4 5 6 7 8];
            app.RSlider.ValueChangedFcn = createCallbackFcn(app,
@RSliderValueChanged, true);
            app.RSlider.MinorTicks = [];
            app.RSlider.Position = [214 201 252 3];
            app.RSlider.Value = 1;

            % Create RLabel
            app.RLabel = uilabel(app.UIFigure);
            app.RLabel.FontSize = 16;
            app.RLabel.FontWeight = 'bold';
            app.RLabel.FontColor = [1 0 0];
            app.RLabel.Position = [168 191 25 22];
            app.RLabel.Text = 'R';

            % Create GLabel
            app.GLabel = uilabel(app.UIFigure);
            app.GLabel.FontSize = 16;
            app.GLabel.FontWeight = 'bold';
            app.GLabel.FontColor = [0 1 0];
            app.GLabel.Position = [168 131 25 22];
            app.GLabel.Text = 'G';

            % Create BLabel
            app.BLabel = uilabel(app.UIFigure);
            app.BLabel.FontSize = 16;
            app.BLabel.FontWeight = 'bold';
            app.BLabel.FontColor = [0 0 1];
            app.BLabel.Position = [168 72 25 22];
            app.BLabel.Text = 'B';

            % Create Load_Image
            app.Load_Image = uibutton(app.UIFigure, 'push');
            app.Load_Image.ButtonPushedFcn = createCallbackFcn(app,
@Load_ImageButtonPushed, true);
            app.Load_Image.Position = [124 240 100 23];
            app.Load_Image.Text = 'Load Image';

            % Create Quantize
```

```
            app.Quantize = uibutton(app.UIFigure, 'push');
            app.Quantize.ButtonPushedFcn = createCallbackFcn(app,
@QuantizeButtonPushed, true);
            app.Quantize.Enable = 'off';
            app.Quantize.Position = [448 240 100 23];
            app.Quantize.Text = 'Quantize';

            % Create Label_2
            app.Label_2 = uilabel(app.UIFigure);
            app.Label_2.HorizontalAlignment = 'right';
            app.Label_2.Position = [169 132 25 22];
            app.Label_2.Text = '';

            % Create GSlider
            app.GSlider = uislider(app.UIFigure);
            app.GSlider.Limits = [1 8];
            app.GSlider.MajorTicks = [1 2 3 4 5 6 7 8];
            app.GSlider.ValueChangedFcn = createCallbackFcn(app,
@GSliderValueChanged, true);
            app.GSlider.MinorTicks = [];
            app.GSlider.Position = [215 141 252 3];
            app.GSlider.Value = 1;

            % Create Label_3
            app.Label_3 = uilabel(app.UIFigure);
            app.Label_3.HorizontalAlignment = 'right';
            app.Label_3.Position = [169 72 25 22];
            app.Label_3.Text = '';

            % Create BSlider
            app.BSlider = uislider(app.UIFigure);
            app.BSlider.Limits = [1 8];
            app.BSlider.MajorTicks = [1 2 3 4 5 6 7 8];
            app.BSlider.ValueChangedFcn = createCallbackFcn(app,
@BSliderValueChanged, true);
            app.BSlider.MinorTicks = [];
            app.BSlider.Position = [215 81 252 3];
            app.BSlider.Value = 1;

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = uniformQuantGUI

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
```

```
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

Published with MATLAB® R2022a