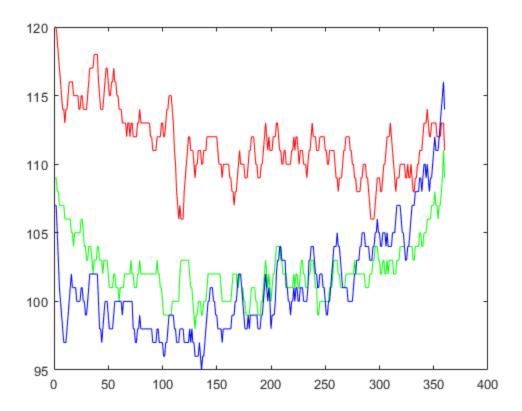
```
% takes an image and scans it for red-blue values
% scans from top to bottom on a layer dependent on input
% then does the same thing with another image
% result is put on a graph to compare, and also given out as percentages
% written by Stephen Xie with help from the internet
mixerStart = imread("WIN 20231011 14 12 43 Pro.jpg");
mixerEnd = imread("WIN_20231011_14_29_30_Pro.jpg");
% for the first image I mostly need the red and blue values
% step one: select section to be scanned for colours
section = mixerStart(571:640, 651:700, :); % inlets merge here
colourLine = section(:, 25, :);
plot(1:70, colourLine(:, :, 1), 'r', 1:70, colourLine(:, :, 2), 'g', 1:70,
colourLine(:, :, 3), 'b');
% step two: derive maximum and minimum values of red and blue
redMax = max(colourLine(:, :, 1));
redMin = min(colourLine(:, :, 1));
redVariance = redMax - redMin;
blueMax = max(colourLine(:, :, 3));
blueMin = min(colourLine(:, :, 3));
blueVariance = blueMax - blueMin;
% step three: repeat with second image at end of micromixer
% note: I cut out the edge bits cos it was blurry
section2 = mixerEnd(61:550,:,:);
colourLine2 = section2(61:420, 1000, :);
plot(1:360, colourLine2(:, :, 1), 'r', 1:360, colourLine2(:, :, 2), 'g',
1:360, colourLine2(:, :, 3), 'b');
% is that pattern noise or actually representative of the result?
% those variations are over really small areas so they're not too relevant
% try filtering out the noise?
colourLine3 = colourLine2;
for i = 3:(length(colourLine2)-2)
    colourLine3(i,:,1) = mean(colourLine2((i-2):(i+2),:,1));
    colourLine3(i,:,2) = mean(colourLine2((i-2):(i+2),:,2));
    colourLine3(i,:,3) = mean(colourLine2((i-2):(i+2),:,3));
end
plot(1:360, colourLine3(:, :, 1), 'r', 1:360, colourLine3(:, :, 2), 'g',
 1:360, colourLine3(:, :, 3), 'b');
% looks like there's still a fair bit of variation but not too noisy
% just eyeballing it, it looks like r varies from 114 to 110
% and b varies from 98 to 112
```

1



Published with MATLAB® R2023a