Udayakumar Mani et al /J. Pharm. Sci. & Res. Vol. 9(6), 2017, 919-925

ISSN:0975-1459
Journal of Pharmaceutical Sciences and Research
www.jpsr.pharmainfo.in

CoMat: An Integrated Tool for Comet Assay Image Analysis

Udayakumar Mani [A] *, Pavithrakumari Manickam [B]

[A] – *Assistant Professor, Department of Bioinformatics, SASTRA University, Thanjavur – 613 401, Tamil Nadu, India.*
[B] – *M.Tech (Integrated), Department of Bioinformatics, SASTRA University, Thanjavur – 613 401, Tamil Nadu, India.*

Abstract

DNA damage analysis plays an important role in determining the approaches for treatment and prevention of various heritable diseases. Comet assay or single-cell gel electrophoresis (SCGE) is a method recognized for studying DNA damage in a single cell. Comet assay images were collected from PubMed by searching with the keyword "Comet Assay". Various steps involved in analyzing the images were removing artifacts, counting comets, labelling comets, partitioning comet into head and tail, calculating area and intensity. Finally the percentage of DNA damage was calculated. We created a standalone tool named "CoMat" for the detection and quantification of DNA damage by processing the comet assay images using MATLAB and VB.Net platform. The user interface was created using VB.Net platform for uploading an image file and saving the calculated parameters in an Excel file. The MATLAB code functions for analyzing the images were written in the MATLAB code editor and a VB.Net extension was created by adding MATLAB function files to Classes under Build .NET Assembly in the MATLAB environment. The created Dynamic-link library (dll) extension was then added as a reference for Visual Basic for executing MATLAB functions through VB.Net. CoMat can process different types of image formats such as JPEG, TIFF, BMP, and PNG. The tool removed unwanted gel artifacts and background noises efficiently. It is also capable of analyzing both silver stained and SYBR green stained images by converting them into grayscale images. It will be able to process only if the image is of higher resolution. The calculated area and intensity values were stored as a cell array in MATLAB which was then exported into an Excel sheet. The results predicted by CoMat were accurate and it was an efficient standalone tool for analyzing DNA damage in a single cell using comet assay images.

Keywords : VB.NET, Matlab, Comet assay, DNA, DLL

INTRODUCTION

Cellular DNA is constantly damaged by chemical agents such as reactive oxygen and nitrogen species (ROS/RNS) and other environmental factors like UV radiation. DNA damage is associated with the etiology of many major diseases. Specifically, oxidative DNA damage has been involved in cardiovascular disease, neuro degenerative diseases and ageing. The single cell gel electrophoresis assay (also known as the Comet assay) is a sensitive method for quantifying DNA strand breaks at the level of the single cell [2]. The basic principle behind comet assay is firstly, the cells are embedded in agarose and lysed, followed by electrophoresis. Upon electrophoresis, undamaged DNA in a supercoiled state remains intact while damaged DNA strand breaks and are relaxed. These damaged DNA migrate to the anode and form a comet shaped structure, which can then be visualized using fluorescence microscopy by staining with a DNA-binding dye. Based on the comet size and shape, the amount of DNA within it needs to be measured in order to assess the level of DNA damage [3]. SCGE is used to study fields involving DNA damage such as environmental toxicology [4], bio-monitoring [5], radiation biology [6], nutritional and cancer studies [7, 8]. The time required for evaluating these images either by visual (manual) scoring or automated software scoring is one of the major drawbacks of comet assay.

In the current study, we have aimed to develop a standalone tool for the quantification of damaged DNA in a single cell. The rationale of this tool is to assess the level of DNA damage within a single cell by processing the microscopic image using MATLAB software. The rationale is to implement a fully automated platform named "CoMat" for the detection and quantification of DNA damage by processing the comet assay images using MATLAB and VB.Net platform.

MATERIALS AND METHODS

Comet assay images were collected from PubMed Images by searching with the keyword "Comet Assay". Various steps involved in analyzing the images were removing artifacts, counting comets, labelling comets, partitioning comet into head and tail, calculating area and intensity and finally calculating percentage DNA damage which is shown in figure 1.

CoMat Implementation:

The user interface was created using VB.Net platform for uploading an image file and saving the calculated parameters in an Excel file. The MATLAB code functions for analyzing the images were written in the MATLAB code editor and a VB.Net extension is created by adding MATLAB function files to Classes under Build .NET Assembly in the MATLAB environment. The created Dynamic-link library (dll) extension was then added as the reference for Visual Basic for executing MATLAB functions through VB.Net. The code for each part of the execution is given below.

Initial declarations and Image Processing Module

The declaration of all the variables is first written in the code editor of Visual Basic's Windows Forms Application (.vb file) and the corresponding Comat GUI is created using Buttons, Labels, PictureBox and OpenFileDialog from the ToolBox present in the designer page of Visual Basic environment.

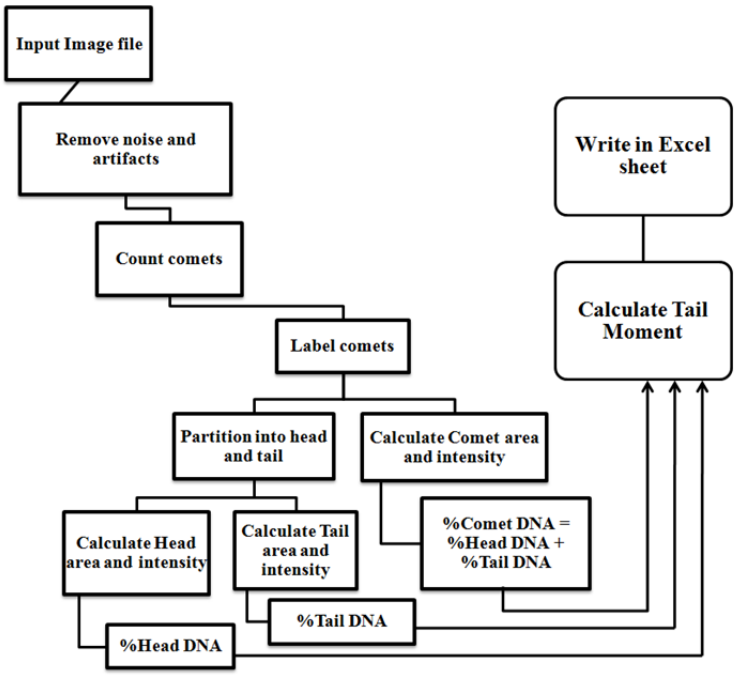

Figure 1: Workflow behind the implementation of CoMat

```
Imports System.Text.RegularExpressions
Imports Excel = Microsoft.Office.Interop.Excel
Public Class Form1
    Dim img As Image
    Dim r1 As Object
    Dim r2 As Object
    Dim ha() As Integer
    Dim path As String
    Dim Matlab As Object
    Dim appXL As Excel.Application
    Dim wbXl As Excel.Workbook
    Dim shXL As Excel.Worksheet
    Dim raXL As Excel.Range
    Dim imgname As String
    Dim out As Object = Nothing
    Dim regex As Regex = New Regex("([A-Za-z0-9\-]+)\.[a-z]+$")
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
        Matlab = CreateObject("Matlab.Application")
        OpenFileDialog1.InitialDirectory = "C:\Users\pavithrakumari\Desktop"
        OpenFileDialog1.Title = "Choose an image for analysis"
        OpenFileDialog1.Filter = "Image file | *.jpg;*.png;*.bmp;*.gif;*.ico|All Files |*.*"
        PictureBox1.SizeMode = PictureBoxSizeMode.Zoom
    End Sub
```

The input image can be uploaded by clicking a button named “Load an Image”. The button calls the OpenFileDialog control for uploading an image file from the user specified directory. Once the user uploads an image, it will appear within the PictureBox in the GUI and the image location is displayed above the PictureBox using the Label option. The image file name alone is extracted for future usage in MATLAB, using the regular expression. The code for the above mentioned tasks is given below:

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
        If OpenFileDialog1.ShowDialog = DialogResult.OK Then
            img = Image.FromFile(OpenFileDialog1.FileName)
            PictureBox1.Image = img
            Label1.Text = OpenFileDialog1.FileName
        End If
        path = OpenFileDialog1.FileName
        Dim match As Match = regex.Match(path)
        If match.Success Then
            imgname = match.Value
        End If
End Sub
```

Calling MATLAB functions from VB.Net

The MATLAB functions can be called from VB.Net after adding reference file (.dll file). The Matlab objects can be created for accessing the functions and Matlab syntaxes. The code for calling the MATLAB function named "pavfun" with three arguments such as the image location, image file name and the flag variable for processing the image is written in the VB.Net editor as mentioned below. The flag variable is either 0 or 1. If the user selects Do Analysis button, the flag variable will hold the value 0 and if the user selects Save Analysis button, the flag variable will hold the value 1. When the user clicks New Analysis button, the previous outputs will be cleared and the GUI is ready for the new analysis.

```
 Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
     Matlab.Execute("cd C:\Users\pavithrakumari\Desktop\fun")
     r2 = Matlab.Execute("pavfun('" + OpenFileDialog1.FileName + "', '" + imgname + "',0)")
     MsgBox(r2)
 End Sub
Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click
        Matlab.Execute("cd C:\Users\pavithrakumari\Desktop\fun")
        r2 = Matlab.Execute("pavfun('" + OpenFileDialog1.FileName + "', '" + imgname + "',1)")
        MsgBox(r2)
 End Sub
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button4.Click
        Matlab.Execute("clc")
        Matlab.Execute("close all")
        Matlab.Execute("clear all")
        PictureBox1.Image = Nothing
        Label1.Text = Nothing
    End Sub
End Class
```

MATLAB MODULES:

The MATLAB function named "pavfun" holds three arguments such as the path of an image, image file name and the flag variable. For both 0 and 1 values of flag variable, the code implements the same process but the difference comes in saving the output file. If the value is 1, the output can be saved as an Excel file (.xlsx file) without displaying the head and tail segmented image. Otherwise, it will do the analysis and just display the head segmented image without saving it as an Excel file.

Reading an image file and initial processing:

The image file can be accessed from the location of an image which is passed as an argument to the function. The first step is to resize the image file to the default size of 500×500 pixels using the imresize inbuilt function. The resized image is then converted into a grayscale image using rgb2gray inbuilt function. Based on the intensity values the comets are identified, the head region is segmented and stored as a binary array for further strategies. The MATLAB code for the above mentioned tasks is given below:

```
function [] = pavfun(imip,imname,flag)
A=imread(imip);
A=imresize(A,[500,500]);
A1=rgb2gray(A);
[r,c]=size(A1);
comim=zeros(r,c);
headim=zeros(r,c);
tim=zeros(r,c);
for i=1:r
    for j=1:c
        if A1(i,j)>48
            comim(i,j)=1;
        end
        if A1(i,j)>120
            headim(i,j)=1;
        end
        if A1(i,j)>43 && A1(i,j)<70
            tim(i,j)=1;
        end
    end
end
```

Counting and Labelling the Comets and Head region:
Based on the area of coverage and intensity, the comet regions and head regions are isolated and the background artifacts are removed using the inbuilt function called "bwareopen". The lower and upper bound threshold values differ for both comet segmentation and head partitioning. After segmenting, calculate centroid for each object using "regionprops" function for labelling them at the center position.

```
comLB = 500;
comUB = 7000;
comB3 = xor(bwareaopen(comim,comLB),  bwareaopen(comim,comUB));
comboundaries = bwboundaries(comB3,'noholes');
lenbou=length(comboundaries);
headLB = 100;
headUB = 2000;
headB3 = xor(bwareaopen(headim,headLB),  bwareaopen(headim,headUB));
headboundaries = bwboundaries(headB3,'noholes');
lenbouh=length(headboundaries);
  comstat = regionprops(comB3,'Centroid');
 headstat = regionprops(headB3,'Centroid');

 if flag==0
figure, imshow(~comB3);title('Labelled comets'); hold on;
 for comk = 1 : lenbou
    comb = comboundaries{comk};
    comc = comstat(comk).Centroid;
    plot(comb(:,2),comb(:,1),'r','linewidth',1);
    text(comc(1),comc(2),num2str(comk),'backgroundcolor','g');
end
hold off;

figure, imshow(~headB3);title('Labelled Head'); hold on;
for headk = 1 : lenbouh
    headb = headboundaries{headk};
    headc = headstat(headk).Centroid;
    plot(headb(:,2),headb(:,1),'r','linewidth',1);
    text(headc(1),headc(2),num2str(headk),'backgroundcolor','g');
end
hold off;
end
```

3.2.3. Calculating Area and Intensity:

```
if flag==1
commeanIntensityValue=zeros(1,lenbou);
hmeanIntensityValue=zeros(1,lenbouh);
% objectMeasurements = regionprops(L, comB3, 'all');
for comk = 1 : lenbou
    comc = comstat(comk).Centroid;
    commeanIntensityValue(comk) = mean2(comc);
end
ca = regionprops(comB3,'Area');
cax=struct2cell(ca);
caxmat=cell2mat(cax);
cmat=caxmat;
comlen= regionprops(comB3,'MajorAxisLength');
comlenx=struct2cell(comlen);
comlenxmat=cell2mat(comlenx);
%To find head information
for headk = 1 : lenbouh
     headc = headstat(headk).Centroid;
     hmeanIntensityValue(headk) = mean2(headc);
end
ha = regionprops(headB3,'Area');
hax=struct2cell(ha);
haxmat=cell2mat(hax);
hlenx=struct2cell(hlen);
hlen= regionprops(headB3,'MajorAxisLength');
hlenxmat=cell2mat(hlenx);
```

Similarly, to calculate area, intensity and major axis length for each object, use the corresponding properties under the function called “regionprops”. The “regionprops” function will return the values in a structure array format, which is converted into a cell array format using “struct2cell” function and finally, cell array to a matrix format using “cell2mat” inbuilt function. The matrix format will be easy for exporting the calculated values into an Excel sheet.

3.2.4. Exporting into Excel file:

The calculated parameters such as CometArea, CometIntensity, CometLength, CometDNA, HeadArea, HeadIntensity, HeadLength, HeadDNA, HeadDNAPercent, TailArea, TailIntensity, TailLength, TailDNA, TailDNAPercent, TailMoment are exported into an Excel file named ‘XXX.xlsx’ where XXX is the file name of the input image.

```
al=length(cmat);
hl=length(haxmat);
if al==hl
expression = '\.(\w+)';
replace = '.xlsx';
xcelname= regexprep(imip,expression,replace);
f2='Num'; f3='CometArea'; f4='CometIntensity'; f5='CometLength'; f6='CometDNA'; f7='HeadArea';
f8='HeadIntensity'; f9='HeadLength'; f10='HeadDNA'; f11='HeadDNAPercent';
f12='TailArea'; f13='TailIntensity'; f14='TailLength'; f15='TailDNA'; f16='TailDNAPercent';
f17='TailMoment';
v2=zeros(1,al); v3=cmat; v4=commeanIntensityValue; v5=comlenxmat; v6=zeros(1,al); v7=haxmat;
v8=hmeanIntensityValue; v9=hlenxmat; v10=zeros(1,al); v11=zeros(1,al);
v12=(cmat-haxmat); v13=commeanIntensityValue-hmeanIntensityValue; v14=comlenxmat-hlenxmat;
v15=zeros(1,al); v16=zeros(1,al); v17=zeros(1,al);
for ij=1:al
    v1(ij)={imname};     v2(ij)=ij;
     comdna=transpose(commeanIntensityValue(ij)*cmat(ij));
        hdna=transpose(hmeanIntensityValue(ij)*haxmat(ij));
        tdna=comdna-hdna;
        kval=(hdna/comdna)*100;
v6(ij)=comdna; v10(ij)=hdna; v11(ij)=kval; v15(ij)=tdna; v16(ij)=(tdna/comdna)*100;
v17(ij)=(comlenxmat(ij)-hlenxmat(ij))*(tdna/comdna);
end
s=struct(f2,v2,f3,v3,f4,v4,f5,v5,f6,v6,f7,v7,f8,v8,f9,v9,f10,v10,f11,v11,f12,v12,f13,v13,f14,v14,f15,v15,f16,v16,
f17,v17);
scell=struct2cell(s);  smat=cell2mat(scell);
% header=fieldnames(s);
header={'FileName','Num','CometArea','CometIntensity','CometLength','CometDNA','HeadArea','HeadIntensity','HeadLe
ngth','HeadDNA','HeadDNAPercent','TailArea','TailIntensity','TailLength','TailDNA','TailDNAPercent','TailMoment'}
;
% xlswrite(xcelname,{'FileName'},'Sheet1','A1');
xlswrite(xcelname,transpose(v1),'Sheet1','A2');
xlswrite(xcelname,header,'Sheet1');
xlswrite(xcelname,transpose(smat),'Sheet1','B2');
end
end
```

Creating .NET Assembly in MATLAB:

After writing the MATLAB function, build .NET Assembly by creating a ‘test’ class and add the pavfun function’s m file (pavfun.m) to the test class and click the Build option located above. It will generate a test_VB_matlabNative.dll file in the directory.

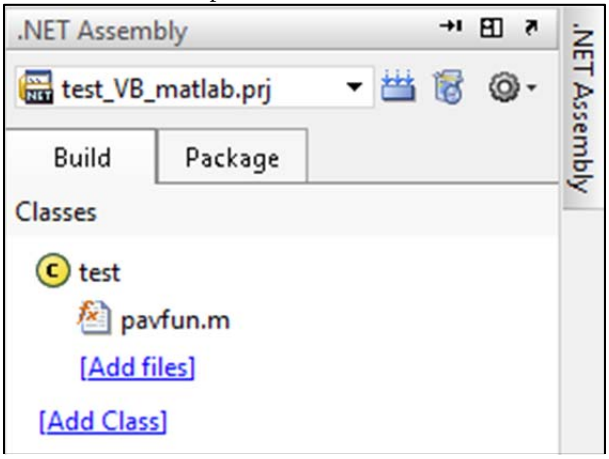

Figure 2: The added m file for the test class under .NET Assembly window in MATLAB

RESULTS

Uploading an image file:

The CoMat's front end for uploading an image file and displaying the uploaded image in the picturebox is as shown in the given figure. Once the image has been uploaded, the location of the image will be displayed above the picturebox which is shown in the figure 3.

Counting and Labelling the Comets and Head region:

When the Do Analysis button is clicked, the image file and its location are passed as arguments to the "pavfun" MATLAB function and the processing of image will take place. The output of the processed image will be the image file without any gel artifacts which is counted and labelled for the number of comets and heads present in it as shown in figure 4.

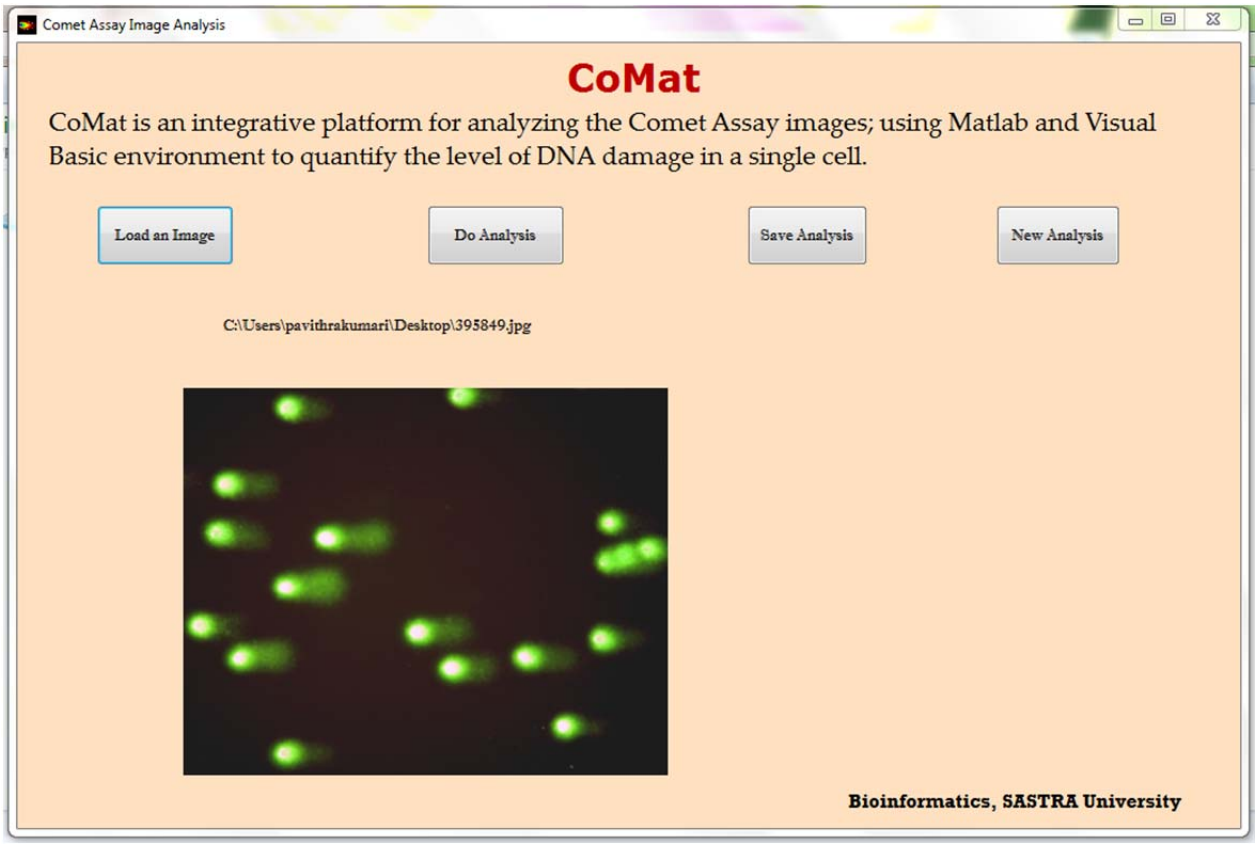

Figure 3: Uploading and displaying an image file

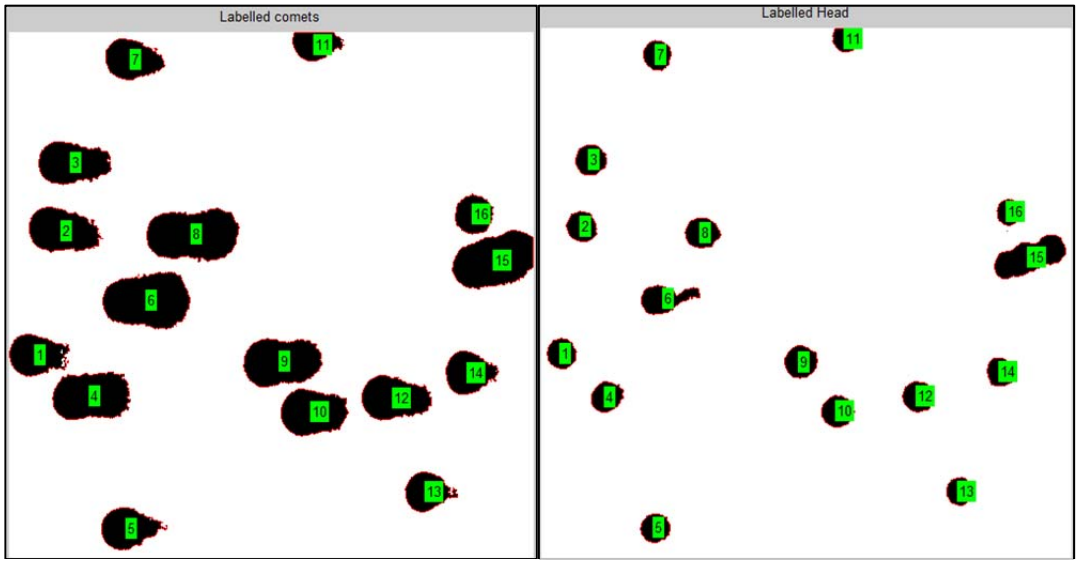

Figure 4: Counting and labelling the total comets and heads

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	FileName	Num	CometAre	CometInt	CometLer	CometDN	HeadArea	HeadInter	HeadLeng	HeadDNA	HeadDNA	TailArea	TailIntens	TailLengt	TailDNA	TailDNAP	TailMoment
2	395849.jp	1	1709	167.4146	57.71402	286111.5	646	163.7461	29.5137	105780	36.9716	1063	3.66844	28.20033	180331.5	63.0284	17.77421
3	395849.jp	2	2266	120.5783	71.38822	273230.5	669	113.3969	30.44201	75862.5	27.76502	1597	7.181471	40.94621	197368	72.23498	29.57748
4	395849.jp	3	2212	93.19372	72.50562	206144.5	693	86.16595	30.11469	59713	28.96657	1519	7.027771	42.39093	146431.5	71.03343	30.11173
5	395849.jp	4	2834	213.0101	78.87426	603670.5	695	205.4532	32.18857	142790	23.65363	2139	7.556819	46.68569	460880.5	76.34637	35.64283
6	395849.jp	5	1667	293.4283	57.30027	489145	624	290.2596	28.88176	181122	37.02828	1043	3.168699	28.41851	308023	62.97172	17.89562
7	395849.jp	6	3590	194.4862	86.95023	698205.5	985	186.8	57.42965	183998	26.35299	2605	7.686212	29.52057	514207.5	73.64701	21.74102
8	395849.jp	7	1644	72.45803	56.57261	119121	605	68.15041	28.82088	41231	34.6127	1039	4.307616	27.75172	77890	65.3873	18.1461
9	395849.jp	8	3583	184.896	93.90945	662482.5	759	173.139	33.29293	131412.5	19.83637	2824	11.75704	60.61651	531070	80.16363	48.5924
10	395849.jp	9	2805	287.3626	78.74857	806052	770	280.4597	31.91576	215954	26.79157	2035	6.902827	46.83281	590098	73.20843	34.28556
11	395849.jp	10	2265	326.3051	66.3241	739081	750	321.2073	31.63332	240905.5	32.59528	1515	5.097744	34.69078	498175.5	67.40472	23.38322
12	395849.jp	11	1053	152.8742	48.78847	160976.5	517	149.6373	27.5918	77362.5	48.05826	536	3.236838	21.19667	83614	51.94174	11.00992
13	395849.jp	12	2168	358.3203	69.16069	776838.5	699	352.2282	30.37305	246207.5	31.69352	1469	6.092158	38.78764	530631	68.30648	26.49447
14	395849.jp	13	1283	418.5475	46.22666	536996.5	550	416.4427	27.03542	229043.5	42.6527	733	2.104818	19.19124	307953	57.3473	11.00566
15	395849.jp	14	1504	381.8537	48.75595	574308	586	379.4403	28.44739	222352	38.71651	918	2.41345	20.30856	351956	61.28349	12.4458
16	395849.jp	15	3268	340.705	84.69537	1113424	1731	338.9532	76.30642	586728	52.69583	1537	1.751812	8.388949	526696	47.30417	3.968323
17	395849.jp	16	1090	308.9505	37.84483	336756	459	307.951	25.44668	141349.5	41.97386	631	0.999478	12.39815	195406.5	58.02614	7.194169

Figure 5: Excel file showing the calculated parameters

Calculating parameters and save as an excel file:

If the user clicks the Save Analysis button, it will do the same task as above but without displaying the processed images. In addition, it will save the calculated parameters in an Excel file named as "image_file_name.xlsx". In the Excel file, the values will be displayed as shown in figure 5

DISCUSSION

CoMat can handle different types of image formats such as JPEG, TIFF, BMP and PNG. The tool removed unwanted gel artifacts and background noises efficiently. CoMat is capable of analyzing both silver stained and SYBR green stained images by converting them into grayscale images. The calculated area and intensity values were stored as a cell array in MATLAB, which was then exported into an Excel sheet.

CONCLUSION

CoMat is an accurate and efficient standalone tool for analyzing DNA damage in a single cell using comet assay images. The average time required to analyze an image using the proposed method was 5-10 seconds depending on the number of comets present in the image. CoMat is capable of quantifying DNA damage in a statistically unbiased manner.

REFERENCES

1) D.Trachootham, W.Lu, M.A.Ogasawara, ValleNR-D, P.Huang, Redox regulation of cell survival, Antioxid.RedoxSignal. 10(2008)1343–1374.
2) A.R.Collins, E.Horvathova, Oxidative DNA damage, antioxidants and DNA repair: applications of the comet assay, Biochem.Soc.Trans. 29(2001)337.
3) O.Ostling, K.J.Johanson, Micro electrophoretic study of radiation-induced DNA damages in individual mammalian cells, Biochem.Biophys.Res.Commun.123 (1984)291–298.
4) A.N.Jha, Ecotoxicological applications and significance of the comet assay, Mutagenesis 23(2008)207–221.
5) M.Dusinska, A.R.Collins, The comet assay in human biomonitoring: gene–environment interactions, Mutagenesis 23(2008)191–205.
6) P.L.Olive, Impact of the comet assay in radio biology, Mutat.Res.681(2009)13–23.
7) G.R.Wasson, V.J.McKelvey-Martin, C.S.Downes, The use of the comet assay in the study of human nutrition and cancer, Mutagenesis 23(2008)153–162.
8) D.J.McKenna, S.R.McKeown, V.J.McKelvey-Martin, Potential use of the comet assay in the clinical management of cancer, Mutagenesis 23(2008)183–190.