```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Video Processing Program GUI%%
%%First Step%%%%%%%%%%%%%%%%%%%%
%%Programed by:%%%%%%%%%%%%%%%%%
%%Ahmad Mohammed Sebaq%%%%%%%%%%
%%Yousra Mohammed Manna%%%%%%%%%
%%19/02/2018%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% In this project - created by us students of the 3rd year of BME - a
 video
% is read by MATLAB GUI, we process the video to access its frames one
 by one.
% We are able to delete the repeated frames
% and create another new video without frame repitition with the same
 frame
% rate of the input video.

function varargout = project(varargin)
% PROJECT MATLAB code for project.fig
%      PROJECT, by itself, creates a new PROJECT or raises the
 existing
%      singleton*.
%
%      H = PROJECT returns the handle to a new PROJECT or the handle
 to
%      the existing singleton*.
%
%      PROJECT('CALLBACK',hObject,eventData,handles,...) calls the
 local
%      function named CALLBACK in PROJECT.M with the given input
 arguments.
%
%      PROJECT('Property','Value',...) creates a new PROJECT or raises
 the
%      existing singleton*.  Starting from the left, property value
 pairs are
%      applied to the GUI before project_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property
 application
%      stop.  All inputs are passed to project_OpeningFcn via
 varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
 only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
```

```
% Edit the above text to modify the response to help project

% Last Modified by GUIDE v2.5 18-Feb-2018 18:41:52

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @project_OpeningFcn, ...
                   'gui_OutputFcn',  @project_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before project is made visible.
function project_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to project (see VARARGIN)

% Choose default command line output for project
handles.output = hObject;

% Create a global VideoReader object(inputVideo)
global inputVideo;
inputVideo = 0;
% Create a global variable (frameNumber)
global frameNumber;
frameNumber = 1;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes project wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = project_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in openButton.
function openButton_Callback(hObject, eventdata, handles)
% Invoke the global VideoReader
global inputVideo;
[name, path] = uigetfile({'*.mp4'; '*.avi'; '*.mkv'}); % get the file
 name and path
% Check for the file existance
if ~isequal(name, 0) || ~isequal(path, 0)
    fullname = fullfile(path, name); % fullnaem = path + name
    inputVideo = VideoReader(fullname); % read the video file
    set(handles.slider1, 'SliderStep', [1/inputVideo.NumberOfFrames ,
 10/inputVideo.NumberOfFrames ]);
    set(handles.slider1, 'Value', 1/inputVideo.NumberOfFrames);
    % Show the 1st frame in the canvas and update the text
    imshow(read(inputVideo, 1));
    axes(handles.axes1);
    text = sprintf('Frame %d of %d', 1, inputVideo.NumberOfFrames);
    set(handles.text2, 'String', text);
else
    % Create error message dialog box
    msgbox('User does not choose a vedio file','Error','error');
end

% --- Executes on button press in removeButton.
function removeButton_Callback(hObject, eventdata, handles)
% Invoke the global VideoReader
global inputVideo;
% Handle the empty VideoReader case(inputVideo)
if ~isequal(inputVideo, 0)
    % Set a counter for the processed videos
    persistent counter;
    if isempty(counter)
        counter = 1; % i for output video number
    else
        counter = counter + 1;
    end

    % Show that gui in process by creating the mouse loading circle
    greenBar = waitbar(0,'Please wait...');
    set(handles.figure1, 'pointer', 'watch')
    drawnow;
    % Create new video with the image sequence
    filename = [sprintf('output%04d',counter) '.avi'];
    outputVideo = VideoWriter(filename);
    % Assign the input video frame rate to the output video frame rate
    outputVideo.FrameRate = inputVideo.FrameRate;
    open(outputVideo) % open the output video to put frame into it
    % Assign the unrepeated frames to the output video
```

```
    writeVideo(outputVideo,read(inputVideo, 1))
    % loop through the input video frames to check for repetition and
    % remove it, the assign the unique frames to the new output video
    steps = inputVideo.NumberOfFrames - 1;
    j = 1; % counter for image naming
    for i = 1 : inputVideo.NumberOfFrames - 1
       img = read(inputVideo, i);
       img2 = read(inputVideo, i+1);
       [m, n, o] = size(img);
       % check the ratio of the difference between a frame and the
 next one
       if (length(find((img == img2)==0)) / (m*n*o)) < 0.07 ||
 length(find(img == 102)) >= 10000
           continue;
       else
           writeVideo(outputVideo,img)
           imageName = [sprintf('%03d',j) '.jpg'];
           imwrite(img,imageName);
           j = j+1;
       end
       waitbar(i / steps);
    end
    % Show that gui in finished
    set(handles.figure1, 'pointer', 'arrow')
    % Finalize the video file.
    close(outputVideo)
    set(handles.slider1, 'SliderStep', [1/inputVideo.NumberOfFrames ,
 10/inputVideo.NumberOfFrames ]);
    set(handles.slider1, 'Value', 1/inputVideo.NumberOfFrames);
    % Read the output video and show it on the canvas and update the
 text
    inputVideo = VideoReader(filename);
    imshow(read(inputVideo, 1));
    axes(handles.axes1);
    text = sprintf('Frame %d of %d', 1, inputVideo.NumberOfFrames);
    set(handles.text2, 'String', text);
    % Create success message dialog box
    msgbox('Operation Completed','Success');
    close(greenBar)
else
    % Create error message dialog box
    msgbox('There is no vedio file','Error','error');
end

% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% Invoke the global VideoReader
global inputVideo;
global frameNumber;
% Handle the empty VideoReader case(inputVideo)
if ~isequal(inputVideo, 0)
    value = get(hObject,'Value');
    % Handle the zero index of the slider
    if ~isequal(value, 0)
```

```
        % Map the slider value to the number of frames
        % and change the shown frame
        percentage = value/1;
        frameNumber = round(percentage*inputVideo.NumberOfFrames);
        % Show the 1st frame in the canvas and update the text
        imshow(read(inputVideo, frameNumber));
        axes(handles.axes1);
        text = sprintf('Frame %d of %d', frameNumber,
 inputVideo.NumberOfFrames);
        set(handles.text2, 'String', text);
    else
        % Show the 1st frame in the canvas and update the text
        imshow(read(inputVideo, 1));
        axes(handles.axes1);
        text = sprintf('Frame %d of %d', 1,
 inputVideo.NumberOfFrames);
        set(handles.text2, 'String', text);
    end
else
    % Create error message dialog box
    msgbox('There is no vedio file','Error','error');
end


% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
 called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
 get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

Published with MATLAB® R2017a