Apéndice D: Código en Matlab de la simulación de la FCICT con algoritmo de compresión de Huffman.

En este apéndice se muestra el código implementado en Matlab para simular la técnica FCICT con algoritmo de compresión de Huffman en la interfase visual.

```
% este código se crea automáticamente para inicializar los valores de la interfase
% gráfica

function varargout = fakehuffmanGUI(varargin)


gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @fakehuffmanGUI_OpeningFcn, ...
                   'gui_OutputFcn',  @fakehuffmanGUI_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
% finaliza el código de inicialización


% el código siguiente llama a las imágenes JPG que se muestran en la interfase

set(hObject, 'Units', 'pixels');
handles.FCICT_GUI = imread(['FCICT_GUI.jpg']); % Read the image file banner.jpg
info = imfinfo(['FCICT_GUI.jpg']); % Determine the size of the image file
position = get(hObject, 'Position');

axes(handles.axes1);
image(handles.FCICT_GUI)
set(handles.axes1, ...
   'Visible', 'off', ...
   'Units', 'pixels', ...
   'Position', [50 30 info.Width info.Height]);

set(hObject, 'Units', 'pixels');
handles.interrogacion = imread(['interrogacion.jpg']); % Read the image file banner.jpg
info = imfinfo(['interrogacion.jpg']); % Determine the size of the image file
position = get(hObject, 'Position');

axes(handles.axes2);
image(handles.interrogacion)
set(handles.axes2, ...
   'Visible', 'off', ...
```

```
   'Units', 'pixels', ...
   'Position', [650 30 info.Width info.Height]);


% estas funciones crean el cuadro de texto en la interfase donde se recibe
% la longitud de la llave K y se almacena en la variable long_K
function long_K_Callback(hObject, eventdata, handles)

handles.long_K = str2double(get(hObject,'String'));
guidata(hObject, handles); % Save the updated structure

function long_K_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se recibe la
%cadena de texto ingresada por el usuario y se almacena en la variable plain_text
function datos_Callback(hObject, eventdata, handles)

handles.plain_text = get(hObject,'String')
guidata(hObject, handles); % Save the updated structure

function datos_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestra el texto
% original en formato ASCII
function textorigASCII_Callback(hObject, eventdata, handles)

function textorigASCII_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestra la secuencia
% de llaves generada
function llaves_enc_Callback(hObject, eventdata, handles)

function llaves_enc_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se mostrara
```

```
% el cifrado RC4
function cifrado_normal_Callback(hObject, eventdata, handles)

function cifrado_normal_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestra el cifrado FCICT
% en el transmisor
function cifrado_fake_Callback(hObject, eventdata, handles)

function cifrado_fake_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestran
% las llaves de encriptación en el receptor
function llaves_decript_Callback(hObject, eventdata, handles)

function llaves_decript_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestra el cifrado FCICT
% en el receptor
function ciffcict_Callback(hObject, eventdata, handles)

function ciffcict_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto que mostrara el cifrado RC4
% en el receptor
function cifradonormal_dec_Callback(hObject, eventdata, handles)

function cifradonormal_dec_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
```

```
% estas funciones crean el cuadro de texto que muestra el texto a encriptar
% en format ASCII en la parte del intruso
function textrecuperado_ASCII_Callback(hObject, eventdata, handles)

function textrecuperado_ASCII_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto donde se muestra el texto
% recuperado por el intruso
function Texto_recuperado_Callback(hObject, eventdata, handles)

function Texto_recuperado_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto que muestra el cifrado FCICT
% recuperado por el intruso
function cifradoFCICT_int_Callback(hObject, eventdata, handles)

function cifradoFCICT_int_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end


% estas funciones crean el cuadro de texto que muestra las llaves
% recuperadas por el intruso
function llaves_intruso_Callback(hObject, eventdata, handles)

function llaves_intruso_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end




%esta función contiene el código del algoritmo FCICT ejecutado al
%presionar el boton Transmitir

function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
handles.plainASCII = double(handles.plain_text)  %convierte la cadena ingresada
% por el usuario en caracteres ASCII


%inicialización de la llave S, RC4

% long_K es el número de bytes con el que se forma el vector K
% mismo que se repetirá tantas veces como sea necesario
% para llenar el vector temporal T

for i = 0:(handles.long_K-1)
   K(i+1) = i;           %generación del vector K
end

S = [0:255];

for j=0:255               % se inicializa S con valores de 0 a 255
   y = mod(j,handles.long_K);    % se repite el vector K tantas veces como sea
   T(j+1) = K(y+1);         % necesario para genera el vector T
end


k = 0;
for m=0:255
   z = (k + S(m+1) + T(m+1));
   k = mod(z,256);
   temp = S(m+1);
   S(m+1) = S(k+1);
   S(k+1) = temp;
end


size_plain = size(handles.plainASCII);
handles.size_plain = size_plain(2); % estas instrucciones calculan el valor
% de la longitud del texto ingresado por el usuario


% algoritmo generador de llaves
i=0;
k=0;


for j = 1:handles.size_plain

   q = i+1;
   i = mod(q,256);
   p = k + S(q);
   k = mod(p,256);
   temp = S(i+1);
   S(i+1) = S(k+1);
   S(k+1) = temp;
   r = (S(i+1) + S(k+1));
   t = mod(r,256);
   handles.Keys(j) = S(t+1);
   cipher(j) = bitxor(handles.Keys(j), handles.plainASCII(j)); %esta sentencia
% realiza la operación XOR entre las llaves y el texto original

end
```

```
% el siguiente conjunto de sentencias, envía las cadenas de texto y de llaves
% a los cuadros de texto respectivos en la interfase

set(handles.textorigASCII,'String',num2str(handles.plainASCII))
set(handles.llaves_enc,'String',num2str(handles.Keys))
set(handles.llaves_decript,'String',num2str(handles.Keys))
set(handles.llaves_intruso,'String',num2str(handles.Keys))

%FCICT encriptacion

%inicialización de la llave S, RC4

% long_K es el número de bytes con el que se forma el vector K
% mismo que se repetirá tantas veces como sea necesario
% para llenar el vector temporal T

for i = 0:(handles.long_K_fakekey-1)
   K(i+1) = i;          %generación del vector K
end

fakekey = [0:255];

for j=0:255              % se inicializa S con valores de 0 a 255
   y = mod(j,handles.long_K_fakekey); % se repite el vector K tantas veces como sea
   T(j+1) = K(y+1);      % necesario para genera el vector T
end

% se genera la llave fakekey
k = 0;
for m=0:255
   z = (k + fakekey(m+1) + T(m+1));
   k = mod(z,256);
   temp = fakekey(m+1);
   fakekey(m+1) = fakekey(k+1);
   fakekey(k+1) = temp;
end

count = 0;

% el siguiente ciclo inicializa el cifrado FCICT con los valores del cifrado RC4
for j = 1:handles.size_plain
   FAKEcipher(j) = cipher(j);
end

% el siguiente ciclo ingresa los caracteres falsos en el cifrado RC4
for j = 2:handles.size_plain
   k = j + count;
   FAKEcipher(k) = cipher(j);

   w = mod(k,256);
   fakecheck = mod(cipher(j) + cipher(j-1) + fakekey(w),2^handles.m_prob);
   if (fakecheck == 0)
      xor_uno = bitxor(cipher(j), cipher(j-1));
      fakechar = bitxor(xor_uno, fakekey(w));
      count = count + 1;
      FAKEcipher(j + count) = fakechar;
   end
end

size_fakecipher = size(FAKEcipher);
```

```
size_fakecipher = size_fakecipher(2); %determina el tamaño del vector del
%cifrado FCICT


% el siguiente ciclo es ejecutado por el receptor, el cual recibe el cifrado FCICT,
% identifica los caracteres falsos y los excluye para obtener un cifrado RC4
k = 2;
for j = 2:handles.size_plain-1

    w = mod(j,256);
    fakecheck = mod(cipher_dec(j) + cipher_dec(j-1) + fakekey(w),2^handles.m_prob);
    if (fakecheck == 0)
       cipher_dec(k) = FAKEcipher_dec(j);
       i = i + 1;
    end
    i = i + 1;
    k = k + 1;
end

%las siguientes instrucciones envían los cifrados RC4 y FCICT a los respectivos
% cuados de texto en la interfase gráfica

set(handles.cifrado_normal,'String',num2str(cipher))
set(handles.cifrado_fake,'String',num2str(FAKEcipher))

set(handles.ciffcict,'String',num2str(FAKEcipher))
set(handles.cifradonormal_dec,'String',num2str(cipher_dec))

set(handles.cifradoFCICT_int,'String',num2str(FAKEcipher))

% el siguiente ciclo realiza la operación XOR entre el cifrado y las llaves para
% que el receptor recupere el texto original
for j = 1:handles.size_plain

   recuperado_ascii_rx(j) = bitxor(handles.Keys(j), cipher(j));

end

text_recuperado_rx = char(recuperado_ascii_rx);

%estas instrucciones envían el texto recuperado a los cuadros de texto del
%receptor en la interfase
set(handles.textrecuperado_ASCII,'String',num2str(recuperado_ascii_rx))
set(handles.Texto_recuperado,'String',text_recuperado_rx)
%estas funciones crean el cuadro de texto que recibe la longitude de la llave
% fakekey
function long_fakekey_Callback(hObject, eventdata, handles)

handles.long_K_fakekey = str2double(get(hObject,'String'));

function long_fakekey_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

%las siguientes funciones crean el cuadro de texto en la interfase donde
%se recibe el valor de m dado por el usuario
function valordem_Callback(hObject, eventdata, handles)
```

```
handles.m_prob = str2double(get(hObject,'String'))
guidata(hObject, handles);

function valordem_CreateFcn(hObject, eventdata, handles)
if ispc
   set(hObject,'BackgroundColor','white');
else
   set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end
```