PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

TCS – Laboratório de Programação I

Victor Raphael de Paulo Silva Oliveira

Vitor Alexandre Diniz Souza

Waltton Santana de Carvalho Morais

Engenharia Eletrônica e de Telecomunicações

Turno: Noite

Belo Horizonte

2015

Sumário

Objetivos

Usar a interface gráfica do MATLAB para criar uma versão para desktop de um jogo popular (‘’jogo da velha’’). No qual o programa mostrará ao final quem é o ganhador.

História do jogo

O nome ‘’velha’’ tem origem na Inglaterra. Era jogado nos finais da tarde por mulheres idosas que por não terem mais condições de bordar em razão da fraqueza da visão, jogavam este jogo simples. Porém há registros de sua origem ser ainda mais antiga. Fala-se em tabuleiros escavados na rocha de templos do antigo Egito, que teriam sido feitos por escravos há 3.500 anos.

Fundamentação Teórica

Regras

O tabuleiro é uma matriz de três linhas por três colunas. Cada jogador escolhe uma marcação. Em cada rodada o jogador deve escolher uma posição vazia da matriz e preenchê-la com o seu símbolo. O objetivo é conseguir três marcações iguais em linha horizontal, vertical ou diagonal, e ao mesmo tempo, quando possível, impedir o adversário de ganhar na próxima jogada.

Informações sobre o desenvolvimento do código:

Para elaboração desse código o principal conceito de programação usado foi:

- Estrutura de decisão

O programa foi dividido em várias funções e cada uma é responsável por um aspecto do jogo.

Código na íntegra

-Script ‘’JV’’

Limpa a tela e todas as variáveis.

```
clear;
clc;
gui;
```

-Função ‘’GUI.m’’

É responsável pela entrada de dados e pela interface gráfica. Quando o botão ‘’novo jogo’’ for pressionado a função ‘’iniciarJogo’’ é executada.

```
function varargout = gui(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @gui_OpeningFcn, ...
                   'gui_OutputFcn',  @gui_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

function gui_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

guidata(hObject, handles);

function varargout = gui_OutputFcn(hObject, eventdata, handles)
varargout{1} = handles.output;

function pushbutton2_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 1, 1));

function pushbutton3_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 1, 2));

function pushbutton4_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 1, 3));

function pushbutton5_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 2, 1));

function pushbutton6_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 2, 2));

function pushbutton7_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 2, 3));

function pushbutton8_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 3, 1));

function pushbutton9_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 3, 2));

function pushbutton10_Callback(hObject, eventdata, handles)
guidata(hObject, executarJogada(handles, 3, 3));

function pushbutton11_Callback(hObject, eventdata, handles)
handles = iniciarJogo(handles);
guidata(hObject, handles);
atualizarGUI(handles);
guidata(hObject, handles);

function figure1_CreateFcn(hObject, eventdata, handles)
handles.jogoAtivo = 0;
guidata(hObject, handles);

function figure1_KeyPressFcn(hObject, eventdata, handles)
if isempty(str2num(eventdata.Character))
    if lower(eventdata.Character) == 'n'
        pushbutton11_Callback(hObject, eventdata, handles)
    end
else
```

```
    guidata(hObject, executarJogada(handles,
fix((str2num(eventdata.Character) - 1) / 3) + 1,
mod(str2num(eventdata.Character) - 1, 3) + 1));
end

function pushbutton11_KeyPressFcn(hObject, eventdata, handles)
figure1_KeyPressFcn(hObject, eventdata, handles)
```

-Função ‘’iniciarJogo’’

Coloca valor verdadeiro na variável ‘’jogoAtivo’’, zera a matriz de posições e habilita o primeiro jogador.

```
function [ h ] = iniciarJogo( handles )

handles.jogoAtivo = 1;
handles.posicoes = zeros(3, 3);
handles.jogadorAtual = 1;

h = handles;

end
```

-Função ‘’executarJogada’’

Caso algum botão ou tecla numérica do teclando for pressionado é executada a função “executarJogada’’ que faz os seguintes procedimentos:

-Verifica se a jogada é permitida
-Caso sim ele altera o valor da posição da matriz selecionada pelo respectivo símbolo do jogador
-Verifica situação do jogo
-Atualiza guide
-Verifica e habilita o próximo jogador
-Caso a jogada não for válida uma é exibida uma mensagem por 3 segundos

```
function [ h ] = executarJogada( handles, linha, coluna )

if jogadaEhPermitida(handles, linha, coluna)
    handles.posicoes(linha, coluna) = handles.jogadorAtual;
    handles.jogoAtivo = verificaSituacaoDoJogo(handles);
    atualizarGUI(handles);
    handles.jogadorAtual = proximoJogador(handles.jogadorAtual);
    if handles.jogoAtivo
        set(handles.text2, 'String', sprintf('Jogador atual : jogador
%d', handles.jogadorAtual));
    end
else
    if handles.jogoAtivo
        set(handles.text2, 'String', 'Jogada não permitida!');
```

```
        t = timer('TimerFcn',@(x,y)set(handles.text2, 'String',
sprintf('Jogador atual : jogador %d',
handles.jogadorAtual)),'StartDelay',3);
        start(t);
    end
end

h = handles;

end
```

-Função ‘’EhPermitida’’

Verificar se o jogo está ativo e se a posição selecionada ainda não foi usada.

```
function [ Eh ] = jogadaEhPermitida( handles, linha, coluna )

Eh =  (handles.jogoAtivo == 1) && (handles.posicoes(linha, coluna) ==
0);

end
```

-Função ‘VerificaSituacaoDoJogo’’

Verifica a situação do jogo.Caso houver um vencedor ou tenha dado velha exibe uma mensagem e coloca a variável ‘’JogoAtivo’’ para falso.

```
function [ jogoAtivo ] = verificaSituacaoDoJogo( handles )

jogoAtivo = handles.jogoAtivo;

v = vencedor(handles.posicoes);

if v ~= 0
    jogoAtivo = 0;
    set(handles.text2, 'String', sprintf('O vencedor é o jogador %d',
v));
else
    if deuVelha(handles.posicoes,
proximoJogador(handles.jogadorAtual))
        jogoAtivo = 0;
        set(handles.text2, 'String','Deu velha...');
    end
end

end
```

-Função ‘’vencedor’’

Verifica se na situação atual do jogo existe um vencedor.

```
function [ jogador ] = vencedor( p )
```

```
jogador = 0;

%horizontais
if ((p(1, 1) ~= 0) && p(1, 1) == p(1, 2) &&  p(1, 1) == p(1, 3))
    jogador = p(1, 1);
end
if ((p(2, 1) ~= 0) && p(2, 1) == p(2, 2) &&  p(2, 1) == p(2, 3))
    jogador = p(2, 1);
end
if ((p(3, 1) ~= 0) && p(3, 1) == p(3, 2) &&  p(3, 1) == p(3, 3))
    jogador = p(3, 1);
end

%verticais
if ((p(1, 1) ~= 0) && p(1, 1) == p(2, 1) &&  p(1, 1) == p(3, 1))
    jogador = p(1, 1);
end
if ((p(1, 2) ~= 0) && p(1, 2) == p(2, 2) &&  p(1, 2) == p(3, 2))
    jogador = p(1, 2);
end
if ((p(1, 3) ~= 0) && p(1, 3) == p(2, 3) &&  p(1, 3) == p(3, 3))
    jogador = p(1, 3);
end

%diagonais
if ((p(1, 1) ~= 0) && p(1, 1) == p(2, 2) &&  p(1, 1) == p(3, 3))
    jogador = p(1, 1);
end
if ((p(1, 3) ~= 0) && p(1, 3) == p(2, 2) &&  p(1, 3) == p(3, 1))
    jogador = p(1, 3);
end


end
```

-Função ‘’deuVelha’’

Verifica se na situação atual do jogo ainda existe a probabilidade de algum jogador vencer ou se deu velha.

```
function [ dv ] = deuVelha( posicoes, proximoJogador )

dv = isempty(possiveisVencedores(posicoes, proximoJogador, 1));

end
```

-Função ‘’possiveisVencedores’’

Verifica se quais são os possíveis vencedores

```
function [ vencedores ] = possiveisVencedores( posicoes, jogadorAtual,
paraNoPrimeiro )

vencedorAtual = vencedor(posicoes);
```

```
vencedores = [];

if vencedorAtual ~= 0
    vencedores = vencedorAtual;
else
    sp = size(posicoes);
    for linha = 1:sp(1)
        for coluna = 1:sp(2)
            if ((posicoes(linha, coluna) == 0) && (isempty(vencedores)
|| (~paraNoPrimeiro)))
                np = posicoes;
                np(linha, coluna) = jogadorAtual;
                vencedores = [vencedores, possiveisVencedores( np,
proximoJogador(jogadorAtual), paraNoPrimeiro)];
            end
        end
    end
end
```

-Função ‘’atulizarGUI’’

Atualiza o texto dentro do botão e habilita ou desabilita o botão.

```
function [ ] = atualizarGUI( handles )
set(handles.pushbutton2, 'String', num2XO(handles.posicoes(1, 1)));
set(handles.pushbutton3, 'String', num2XO(handles.posicoes(1, 2)));
set(handles.pushbutton4, 'String', num2XO(handles.posicoes(1, 3)));

set(handles.pushbutton5, 'String', num2XO(handles.posicoes(2, 1)));
set(handles.pushbutton6, 'String', num2XO(handles.posicoes(2, 2)));
set(handles.pushbutton7, 'String', num2XO(handles.posicoes(2, 3)));

set(handles.pushbutton8, 'String', num2XO(handles.posicoes(3, 1)));
set(handles.pushbutton9, 'String', num2XO(handles.posicoes(3, 2)));
set(handles.pushbutton10, 'String', num2XO(handles.posicoes(3, 3)));

set(handles.pushbutton2, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(1, 1) == 0));
set(handles.pushbutton3, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(1, 2) == 0));
set(handles.pushbutton4, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(1, 3) == 0));

set(handles.pushbutton5, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(2, 1) == 0));
set(handles.pushbutton6, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(2, 2) == 0));
set(handles.pushbutton7, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(2, 3) == 0));

set(handles.pushbutton8, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(3, 1) == 0));
set(handles.pushbutton9, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(3, 2) == 0));
```

```
set(handles.pushbutton10, 'Enable', boolToOnOff(handles.jogoAtivo &&
handles.posicoes(3, 3) == 0));

end
```

-Função ‘’boolToOnOff’’

Converte uma variável booleana em uma string cujo conteúdo seja on ou off.

```
function [ onOff ] = boolToOnOff( value )
if value
    onOff = 'On';
else
    onOff = 'Off';
end

end
```

-Função ‘’num2XO’’

Converte o símbolo numérico do jogador em espaço, X ou O.

```
function [ XO ] = num2XO( num )

if num == 0
    XO = '';
elseif num == 1
    XO = 'X';
elseif num == 2
    XO = 'O';
end

end
```

-Função ‘’proximoJogador’’

Determina qual será o próximo jogador e atualiza o símbolo na variável ‘’jogadorAtual’’.

```
function [ jogadorAtual ] = proximoJogador( jogador )

jogadorAtual = jogador + 1;

if jogadorAtual == 3
    jogadorAtual = 1;
end

end
```

Instruções do jogo

Para executar o jogo :

-Descompactar a pasta ‘’Jogo da Velha’’ e adicioná-la ao patch do MATLAB.
-Digitar na janela de comando ‘’jv’’.
-Pressionar a tecla ‘’n’’ ou o botão novo jogo para iniciar uma partida.
-Para jogar utilize os botões gráficos ou o teclado numérico do computador (No teclado numérico cada posição do número representa a posição da matriz)

Características do Jogo:

O programa consegue verificar e exibir quem foi o vencedor. Também verifica se ‘’deu velha’’.

A expressão ‘’deu velha’’ significa que na situação atual do jogo não é possível existir um vencedor. Para continuar jogando é necessário criar uma nova partida.