```
function varargout = Assignment2_M2A(varargin)
% ASSIGNMENT2_M2A MATLAB code for Assignment2_M2A.fig
%      ASSIGNMENT2_M2A, by itself, creates a new ASSIGNMENT2_M2A or raises
the existing
%      singleton*.
%
%      H = ASSIGNMENT2_M2A returns the handle to a new ASSIGNMENT2_M2A or the
handle to
%      the existing singleton*.
%
%      ASSIGNMENT2_M2A('CALLBACK',hObject,eventData,handles,...) calls the
local
%      function named CALLBACK in ASSIGNMENT2_M2A.M with the given input
arguments.
%
%      ASSIGNMENT2_M2A('Property','Value',...) creates a new ASSIGNMENT2_M2A
or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Assignment2_M2A_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Assignment2_M2A_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 Assignment2_M2A

% Last Modified by GUIDE v2.5 15-Apr-2019 16:09:30

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Assignment2_M2A_OpeningFcn, ...
                   'gui_OutputFcn',  @Assignment2_M2A_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 Assignment2_M2A is made visible.
function Assignment2_M2A_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 Assignment2_M2A (see VARARGIN)
handles.vR = 0;
handles.vL = 0;
% Choose default command line output for Assignment2_M2A
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Assignment2_M2A wait for user response (see UIRESUME)
% uiwait(handles.figure1);
clear all;
global a;

a = arduino ("/dev/tty.usbserial-AI03LKO1", "uno");


% --- Outputs from this function are returned to the command line.
function varargout = Assignment2_M2A_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;

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

set(handles.Turn_Left, 'Enable','on');
set(handles.Turn_Right, 'Enable','on');
set(handles.Reset, 'Enable','on');
set(handles.Increase_Speed,'Enable','on');
set(handles.Decrease_Speed,'Enable','on');
cla reset;

guidata(hObject,handles);



function Voltage_output(handles) % creating function to call as the cxode
repeates in each button
x = 0;
z = 0;

    for k=1:.5:100
        axes(handles.LMI);
        x = [x,handles.vL];
        plot(x,'LineWidth',2); grid on;
        axis([0 100 -5 5]);
```

```
        pause(0.01)

        axes(handles.RMI);
        z = [z,handles.vR];
        plot(z,'LineWidth',2); grid on;
        axis([0 100 -5 5]);
        pause(0.01);

        vel = sprintf("Left vel = : %.1f     Right vel = %.1f", handles.vL,
handles.vR);
        set(handles.Display_Speed,"String",vel);
    end


function Turn_Left_Callback(hObject, eventdata, handles)
% hObject    handle to Turn_Left (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;
handles = guidata(hObject);
handles.vR = handles.vR + 0.5;

    if handles.vR >= 5
        handles.vR = 5;
    end

    writeDigitalPin(a, 'D7',1);
    writeDigitalPin(a, 'D8',0);
    writePWMVoltage(a, 'D6',handles.vR);
    guidata(hObject,handles);

    writeDigitalPin(a, 'D2',0);
    writeDigitalPin(a, 'D4',0);
    guidata(hObject,handles);    %update the value and keep in into memory
    Voltage_output(handles);




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

handles = guidata(hObject);
handles.vL = handles.vL+0.5;


if handles.vL >= 5
    handles.vL = 5;
end
writeDigitalPin(a, 'D2',1);
writeDigitalPin(a, 'D4',0);
```

```
writePWMVoltage(a, 'D5',handles.vL);
guidata(hObject,handles);

writeDigitalPin(a, 'D7',0);
writeDigitalPin(a, 'D8',0);
guidata(hObject,handles);

Voltage_output(handles);


% --- Executes on button press in Decrease_Speed.
function Decrease_Speed_Callback(hObject, eventdata, handles)
% hObject    handle to Decrease_Speed (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;

handles = guidata(hObject);

handles.vL = handles.vL - 0.5;
handles.vR = handles.vR - 0.5;


if handles.vL <= -5
    handles.vL = -5;
end
if handles.vR <= -5
    handles.vR = -5;
end

if handles.vL >= 0
    writeDigitalPin(a, 'D2', 1);
    writeDigitalPin(a, 'D4', 0);
    writePWMVoltage(a, 'D5', handles.vL);
    guidata(hObject,handles);
else
    writeDigitalPin(a, 'D2', 0);
    writeDigitalPin(a, 'D4', 1);
    writePWMVoltage(a, 'D5', handles.vL);
    guidata(hObject,handles);
end
if handles.vR >= 0
    writeDigitalPin(a, 'D7', 1);
    writeDigitalPin(a, 'D8', 0);
    writePWMVoltage(a, 'D6', handles.vR);
    guidata(hObject,handles);
else
    writeDigitalPin(a, 'D7', 0);
    writeDigitalPin(a, 'D8', 1);
    writePWMVoltage(a, 'D6', handles.vR);
    guidata(hObject,handles);
end

Voltage_output(handles);
```

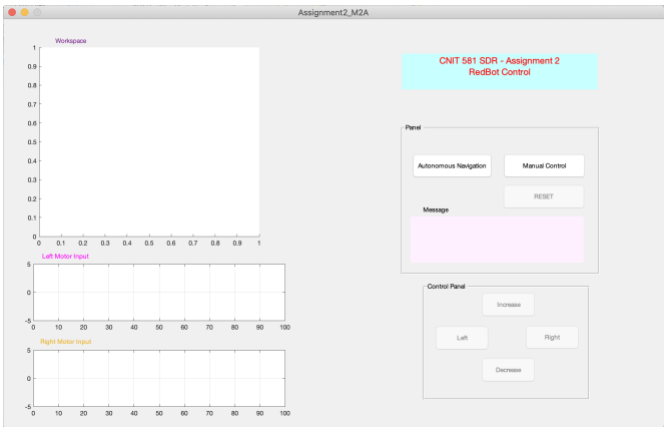
```
%guidata(hObject,handles);

% --- Executes on button press in Turn_Right.
function Increase_Speed_Callback(hObject, eventdata, handles)
% hObject    handle to Increase_Speed (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;

% Retrieve GUI data (the handles structure)
handles = guidata(hObject);

handles.vL = handles.vL + 0.5;
handles.vR = handles.vR + 0.5;


if handles.vL >= 5
    handles.vL = 5;
end
if handles.vR >= 5
    handles.vR = 5;
end

if handles.vL >= 0
    writeDigitalPin(a, 'D2', 1);
    writeDigitalPin(a, 'D4', 0);
    writePWMVoltage(a, 'D5', handles.vL);
    guidata(hObject,handles);   % Update handles structure
else
    writeDigitalPin(a, 'D2', 0);
    writeDigitalPin(a, 'D4', 1);
    writePWMVoltage(a, 'D5', handles.vL);
    guidata(hObject,handles);   % Update handles structure
end
if handles.vR >= 0
    writeDigitalPin(a, 'D7', 1);
    writeDigitalPin(a, 'D8', 0);
    writePWMVoltage(a, 'D6', handles.vR);
    guidata(hObject,handles);   % Update handles structure
else
    writeDigitalPin(a, 'D7', 0);
    writeDigitalPin(a, 'D8', 1);
    writePWMVoltage(a, 'D6', handles.vR);
    guidata(hObject,handles);   % Update handles structure
end

Voltage_output(handles);


%guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
```

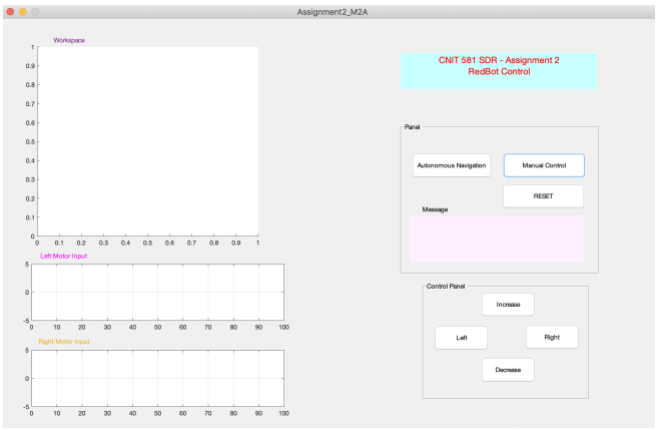
```
%function Left_Motor_Input_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Left_Motor_Input (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


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


handles = guidata(hObject);
handles.vR = 0;
handles.vL = 0;
%guidata(hObject,handles);



writeDigitalPin(a, 'D2', 1);
writeDigitalPin(a, 'D4', 0);
writePWMVoltage(a, 'D5', handles.vL);
guidata(hObject,handles);


writeDigitalPin(a, 'D7', 1);
writeDigitalPin(a, 'D8', 0);
writePWMVoltage(a, 'D6', handles.vR);
guidata(hObject,handles);

Voltage_output(handles);

guidata(hObject,handles);


function Display_Speed_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Display_Speed (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
%global a;

guidata(hObject,handles);
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function LMI_CreateFcn(hObject, eventdata, handles)
% hObject    handle to LMI -Left Motor Input-(see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

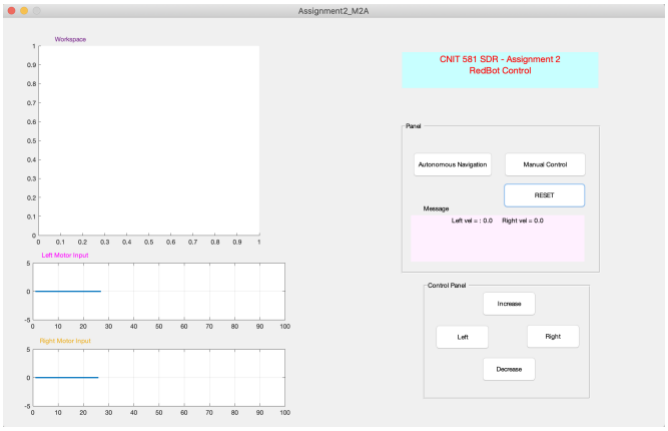
```
% handles    empty - handles not created until after all CreateFcns called;



% Hint: place code in OpeningFcn to populate LMI
% --- Executes during object creation, after setting all properties.
function RMI_CreateFcn(hObject, eventdata, handles)
% hObject    handle to RMI  -Right Motor Input-(see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: place code in OpeningFcn to populate RMI


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

% Hint: place code in OpeningFcn to populate Position

% --- Executes on button press in Autonomous_Drive.
function Autonomous_Drive_Callback(hObject, eventdata, handles)
% hObject    handle to Autonomous_Drive (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global a;


axis([-20 260 -20 260]);
xlabel('X-axis')
ylabel('Y-axis')
grid on

hold on

set(gca,'ydir','reverse');
[rx , ry] = getpts;


C_Robot_Pos = [(rx(2)+rx(1))/2 (ry(2)+ry(1))/2]; % robot's x pos and y pos
C_Robot_Angr = 1*atan2((ry(2)-ry(1)),(rx(2)-rx(1))); % robot's current angle
in radian
C_Robot_Angd = rad2deg(C_Robot_Angr); % robot's current angle in degree
plot([rx(1) rx(2)],[ry(1) ry(2)],'-k','LineWidth',1);


drawbotn([C_Robot_Pos C_Robot_Angr], 5, 1);
hold on
%desired posture
[rx , ry] = getpts;

D_Robot_Pos = [(rx(2)+rx(1))/2 (ry(2)+ry(1))/2]; % robot's x pos and y pos
```

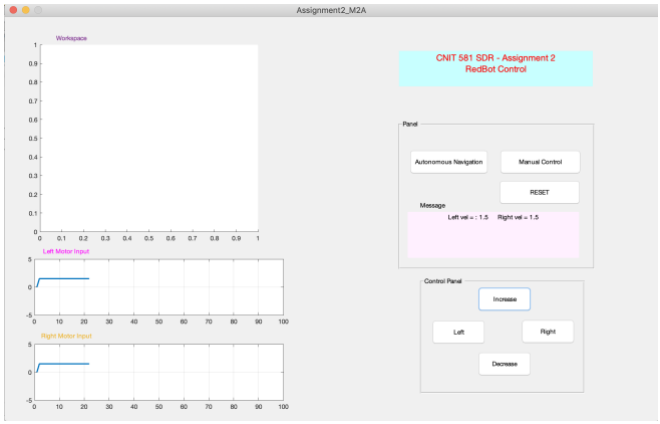
```
D_Robot_Angr = 1*atan2((ry(2)-ry(1)),(rx(2)-rx(1))); % robot's current angle
in radian
D_Robot_Angd = rad2deg(D_Robot_Angr); % robot's current angle in degree
drawbotn([D_Robot_Pos D_Robot_Angr], 5, 1);

dt = .5;                                    %timestep between driving and
collecting sensor data
Tsim = 100;                                  %simulation time
d = 20;                                     %robot's distance

% P controller gains
k_rho = 0.05;                               %should be larger than 0, i.e, k_rho
> 0
k_alpha = 0.8;                              %k_alpha - k_rho > 0
k_beta = -0.008;                            %should be smaller than 0, i.e,
k_beta < 0

vel_data = zeros(length(0:dt:Tsim),2);  %initial vector to analyze velocity
j=1;  %for vel_data vector count




for i = 0:dt:Tsim


    delta_x = D_Robot_Pos(1) - C_Robot_Pos(1);
    delta_y = D_Robot_Pos(2) - C_Robot_Pos(2);
    rho = sqrt(delta_x^2+delta_y^2);    %distance between the center of the
robot's wheel axle and the goal position.
    alpha = -C_Robot_Angr+atan2(delta_y,delta_x); %angle between the robot's
current direction and the vector connecting the center of the axle of the
sheels with the final position.

    %limit alpha range from -180 degree to +180
    if rad2deg(alpha) > 180
        temp_alpha = rad2deg(alpha) - 360;
        alpha = deg2rad(temp_alpha);
    elseif rad2deg(alpha) < -180
        temp_alpha = rad2deg(alpha) + 360;
        alpha = deg2rad(temp_alpha);
    end

    beta = -C_Robot_Angr-alpha;

    % P controller
    v = k_rho*rho;
    w = k_alpha*alpha + k_beta*beta;
    handles.vL = v + d/2*w;
    handles.vR = v - d/2*w;
    %setting limits for left motor & right motoer voltage
    if handles.vL <= -5
        handles.vL = -5;
    end
```

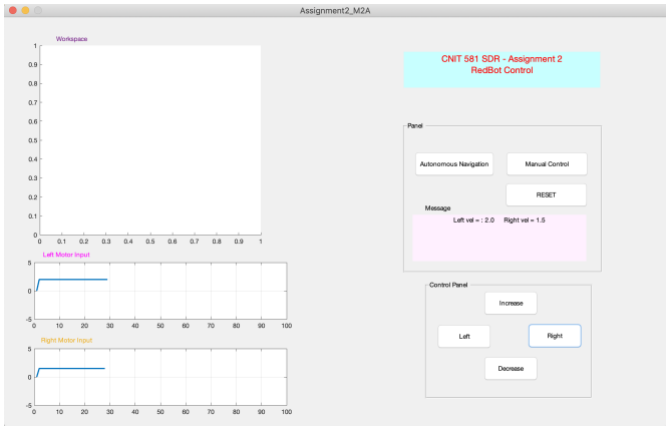
```
    if handles.vR <= -5
        handles.vR = -5;
    end
    if handles.vL >= 5
        handles.vL = 5;
    end

    if handles.vR >= 5
        handles.vR = 5;
    end

    vel_data(j,:) = [handles.vL handles.vR];
    j=j+1;

    posr = [C_Robot_Pos C_Robot_Angr];
    posr = drive(posr, d, handles.vL, handles.vR, dt, posr(3)); %determine
new position
    drawbotn(posr, 5, 1);
    C_Robot_Pos = [posr(1) posr(2)];
    C_Robot_Angr = posr(3);
    pause(0.01);



    vel = sprintf("Left vel = : %.1f     Right vel = %.1f", handles.vL,
handles.vR);
    set(handles.Display_Speed,"String",vel);



    writeDigitalPin(a, 'D2', 1);
    writeDigitalPin(a, 'D4', 0);
    writePWMVoltage(a, 'D5', handles.vL);
    guidata(hObject,handles);

    writeDigitalPin(a, 'D7', 1);
    writeDigitalPin(a, 'D8', 0);
    writePWMVoltage(a, 'D6', handles.vR);



    guidata(hObject,handles);


end
```

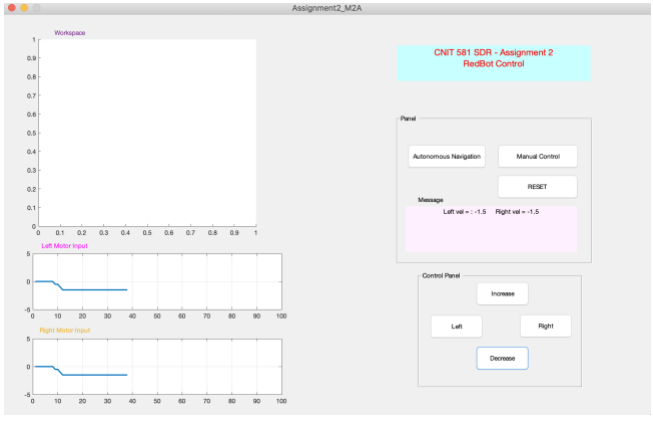

Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Control Panel
Increase
Left
Right
Decrease


Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Control Panel
Increase
Left
Right
Decrease

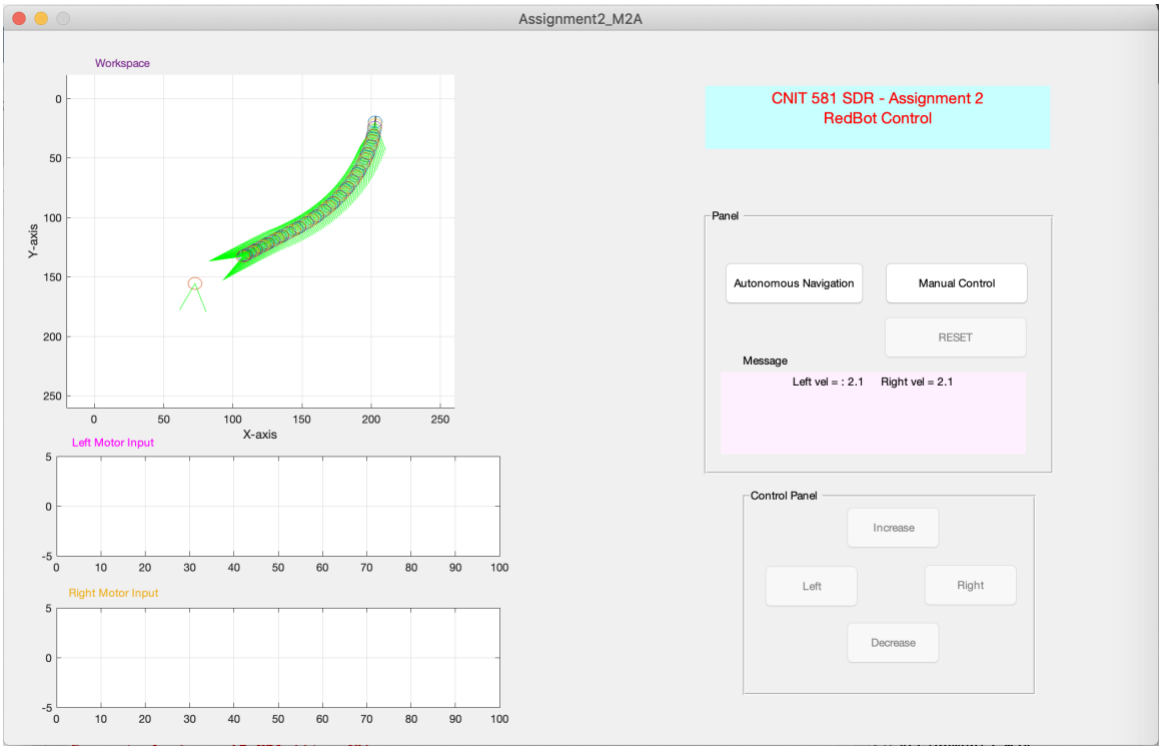

Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Left vel = : 0.0 Right vel = 0.0
Control Panel
Increase
Left
Right
Decrease


Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Left vel = : 1.5 Right vel = 1.5
Control Panel
Increase
Left
Right
Decrease

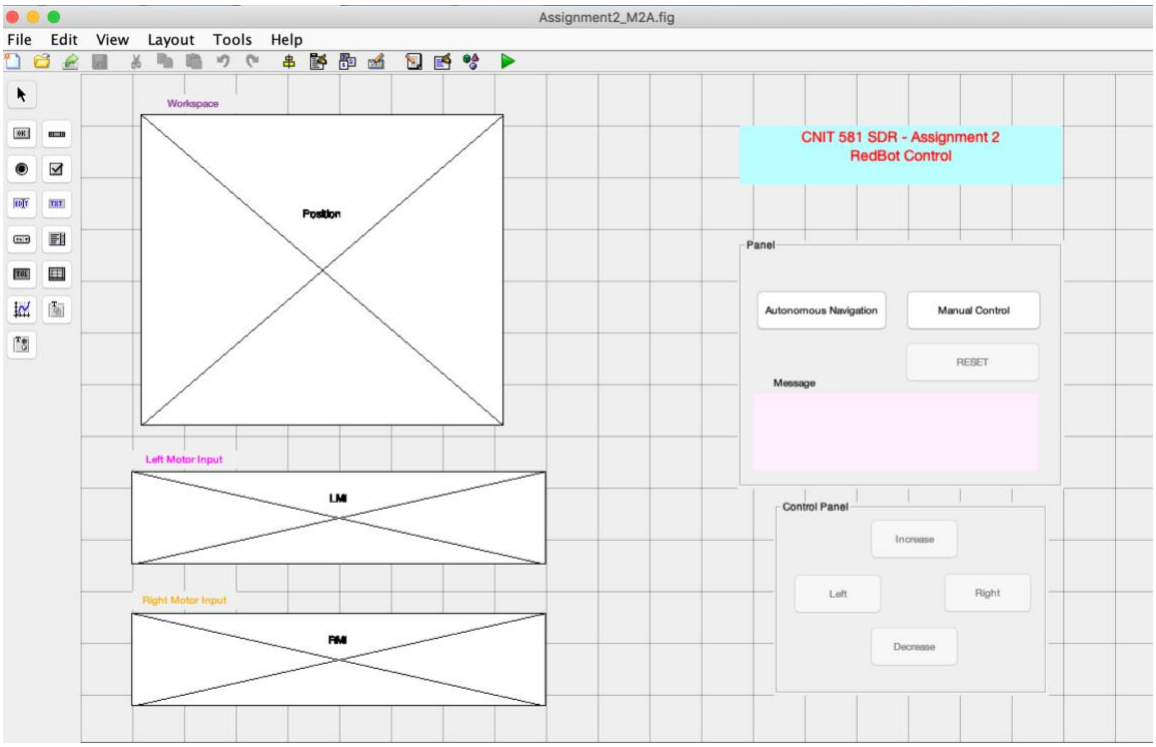

Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Left vel = : 2.0 Right vel = 1.5
Control Panel
Increase
Left
Right
Decrease


Assignment2_M2A
Workspace
Left Motor Input
Right Motor Input
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Left vel = : -1.5 Right vel = -1.5
Control Panel
Increase
Left
Right
Decrease

Manual Drive Different functions in GUI



Autonomous Drive Different functions in GUI


Assignment2_M2A.fig
File Edit View Layout Tools Help
Workspace
Position
Left Motor Input
LMI
Right Motor Input
RMI
CNIT 581 SDR - Assignment 2
RedBot Control
Panel
Autonomous Navigation
Manual Control
RESET
Message
Control Panel
Increase
Left
Right
Decrease

GUI outline