Faculty of Power and Aeronautical Engineering

Department of Robotics

Mobile Robot Report

Task 1

Jafar Jafar,

Pratama Aji Nur Rochman

supervised by

dr. Wojciech Szynkiewicz, & Dawid Seredyński

Contents

1 Introduction

The main goal of the first tutorial is to learn to generate simple trajectory for a mobile robot. The robot should follow a desired trajectory, i.e. move along a straight line to a commanded target point and reach both the commanded position and orientation.

The trajectory is a straight line to the specified destination point represented as three coordinates: [x, y, phi], where x and y is the position of the robot in the XY plane, and phi is the orientation of the robot. You need to provide the youBot with a program that allows it to move along the straight-line trajectory from the starting point (any starting point is possible) to the user-defined destination point.

The commanded orientation (phi) may be reached before or after reaching the desired position (x,y). However, the orientation should change during the linear motion:

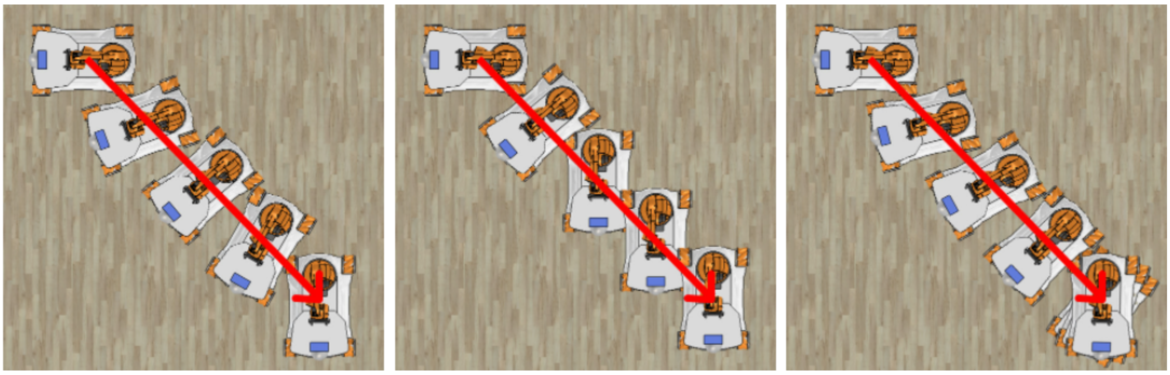

Figure 1: Orientation position

2 Task requirements

- The name of the control callback function should be solution1.
- The motion of the robot must be generated by P regulators with limited output - for (x,y) and for phi
- Both position and orientation must be controlled The robot must follow the shortest path: the straight line from the current position to the destination point
- When the destination point is reached, the function must return finish=true to stop the control program
- You can use the environment file exercise01.ttt

Please Note that the regulator of angular velocity and regulators of linear velocities are independent, so the following cases are acceptable:

- The robot reaches the desired orientation before reaching the desired point (as in the central picture)

- The robot reaches the desired point before reaching the desired orientation (as in the picture on the right)

Still, at the end, both desired orientation and position must be reached within some tolerance, e.g. 0.05 meters for position and 5 degrees for orientation.

3 Solution

3.1 Code

The code was named solution1.m and located in youbot directory. The callback function for this task was declared as:

```
function [forwBackVel, leftRightVel, rotVel, finish] = solution1(pts, contacts, position,
     orientation, varargin)
% The control loop callback function — the solution for task 1A

    % get the destination point
    if length(varargin) ~= 3,
        error('Wrong number of additional arguments: %d\n', length(varargin));
    end
    dest_x = varargin{1};
    dest_y = varargin{2};
    dest_phi = varargin{3};

    % declare the persistent variable that keeps the state of the Finite
    % State Machine (FSM)
    persistent state;
    global fBVel;
    global lRVel;
    if isempty(state),
        % the initial state of the FSM is 'init'
        state = 'init';
    end

    % initialize the robot control variables (returned by this function)
    finish = false;
```

```
    forwBackVel = 0;
    leftRightVel = 0;
    rotVel = 0;
    persistent init_position;
    persistent init_orientation;

    % TODO: manage the states of FSM
    % Write your code here...
    if strcmp(state, 'init'),
        state = 'move';
        init_position = position;
        init_orientation = orientation(3);
    elseif strcmp(state, 'move'),
        forwBackVel = dest_y;
        leftRightVel = dest_x;
        rotVel = 0;
        %control velocity
        vel_control = sqrt(forwBackVel^2+leftRightVel^2);
        vel_max = 1;
        if vel_control > vel_max
            f = vel_max / vel_control;
            vel_lim = f * [forwBackVel;leftRightVel];
            lRVel = vel_lim(1);
            fBVel = vel_lim(2);
        elseif vel_control < vel_max
            lRVel = forwBackVel;
            fBVel = leftRightVel;
        end
        %control angular velocity
        error_phi = dest_phi - orientation(3);
        u_phi = 0.8 * error_phi;
        if u_phi > 1
            u_phi = 1;
        elseif u_phi < 1
            u_phi = .5;
```

```
        end
        %frame transformation
        theta = orientation(3);
        R_GlobalToLocal = [cos(theta),-sin(theta);sin(theta),cos(theta)];
        R_LocalToGlobal = R_GlobalToLocal';
        vel_global = [fBVel; lRVel];
        vel_local = R_LocalToGlobal * vel_global;
        forwBackVel = vel_local(2);
        leftRightVel = vel_local(1);
        rotVel = u_phi;
        %convert phi into 360
        phi_conv = orientation(3) - init_orientation;
        if phi_conv < -3
            phi_conv = phi_conv + (2*pi);
        elseif phi_conv > 3
            phi_conv = phi_conv - (2*pi);
        end
        %stop condition 1, rotation finish first
        if phi_conv > dest_phi
            rotVel=0;
            if (position(1) > init_position(1) + dest_x) && (position(2) > init_position
                (2) + dest_y)
                state = 'finish';
            end
        end
        %stop condition 2, translation finish second
        if (position(1) > init_position(1) + dest_x) && (position(2) > init_position(2) +
             dest_y)
            forwBackVel = 0;
            leftRightVel = 0;
            if phi_conv >  dest_phi
                state = 'finish';
            end
        end
        %stop program
```

```
    elseif strcmp(state, 'finish'),
        finish = true;
        fprintf('X= %f  Y= %f Phi= %d', position(1) - init_position(1), position(2) -
            init_position(2));
        disp('finished');
    else
        error('Unknown state %s.\n', state);
    end
end
```

To run the simulation and the control program using with the callback control function solution1. The destination point is passed to the control program through the variable-length argument list in the run simulation function, e.g. run simulation(@solution1, false, 1, 2, deg2rad(90)) where the arguments are:

- solution1 is the name of the control callback function,
- false - do not display the sensor data (simulation runs faster),
- 1.0 - the x coordinate of the destination point is 1 meter,
- 2.0 - the y coordinate of the destination point is 2 meters,
- deg2rad(90) - the phi coordinate of the destination point (in radians, thus the conversion deg2rad is needed if you prefer to use degrees).

4 Result

The figure below shows that the robot can follow a straight line trajectory and reach the destination point along the X and Y directions with desired orientation.

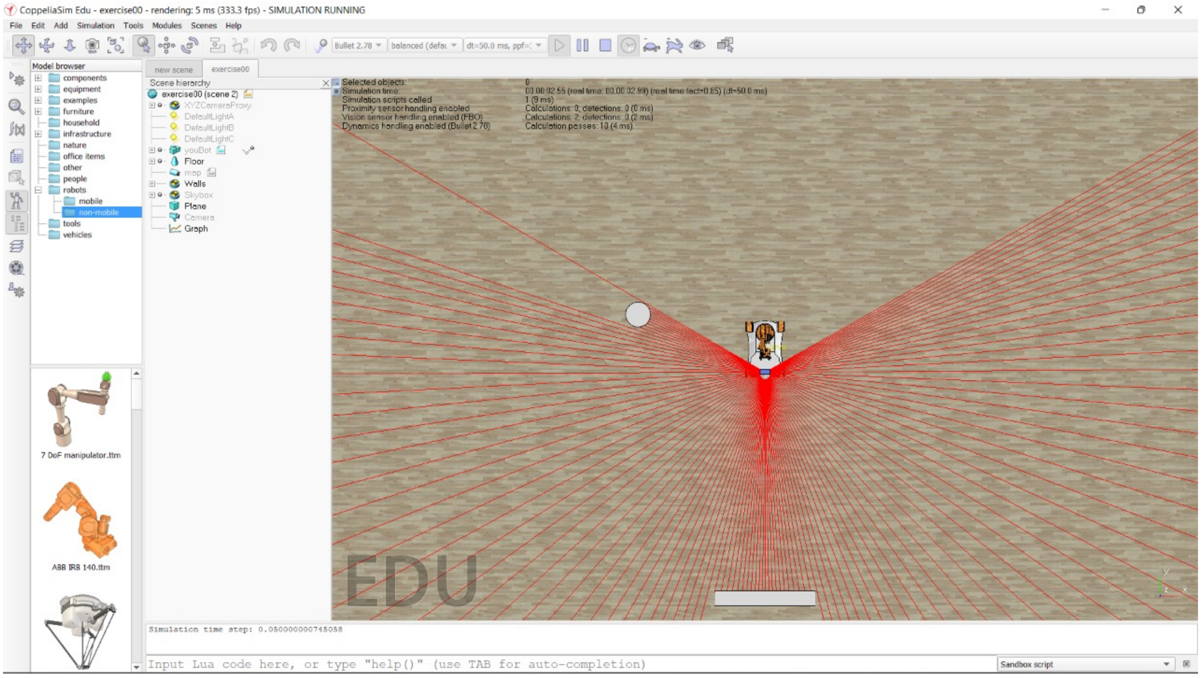

Figure 2: Starting point

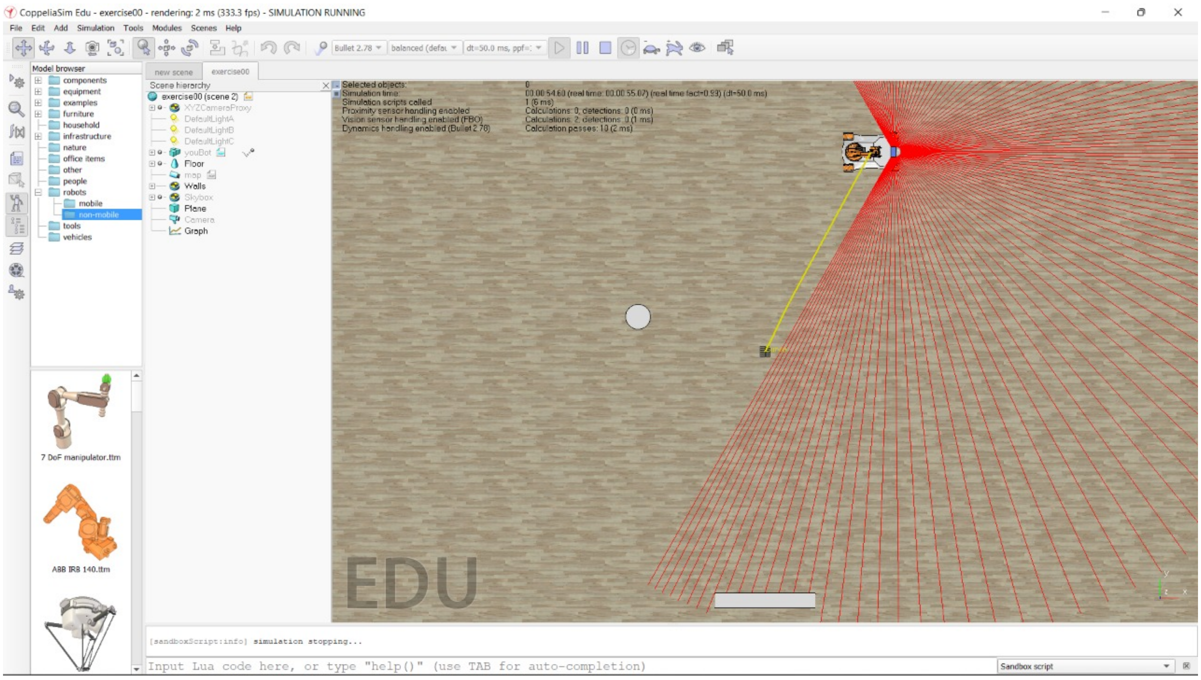

Figure 3: Destination point

At the destination point the final value was obtained X= 1.048488 Y= 2.000232