

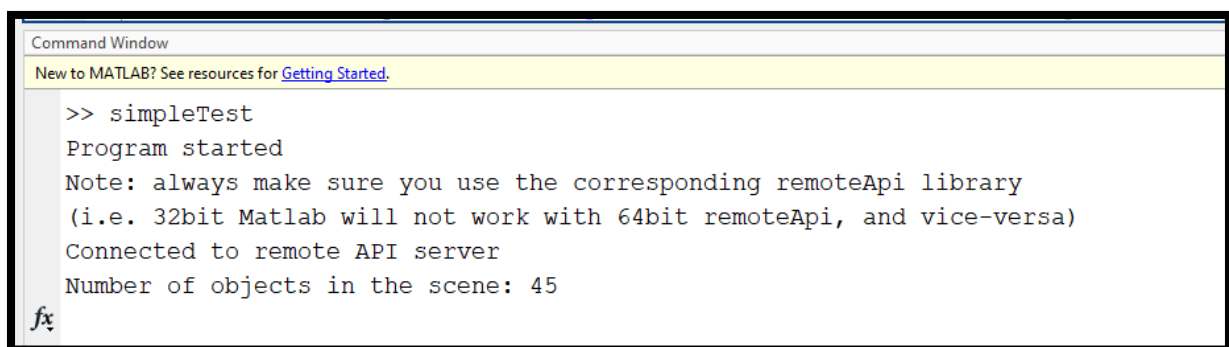
# Sprawozdanie Lab 12

---

Andrzej Żaba, gr\_lab 4, nr indeksu: 401490

Stworzono przestrzeń roboczą. Skopiowano do niej potrzebne pliki. Utworzono scenę w programie V-rep do której skopiowano robota *bubbleRob* stworzonego na pierwszych zajęciach.

Uruchomienie testowego skryptu powiodło się bez problemów.

A screenshot of the MATLAB Command Window. The title bar says 'Command Window'. Below it, a yellow banner reads 'New to MATLAB? See resources for [Getting Started.](#)'. The command prompt shows the execution of 'simpleTest', which outputs 'Program started', a note about using the correct remoteApi library (32bit vs 64bit), 'Connected to remote API server', and 'Number of objects in the scene: 45'. A small 'fx' icon is visible at the bottom left of the window.

```
Command Window
New to MATLAB? See resources for Getting Started.

>> simpleTest
Program started
Note: always make sure you use the corresponding remoteApi library
(i.e. 32bit Matlab will not work with 64bit remoteApi, and vice-versa)
Connected to remote API server
Number of objects in the scene: 45
fx
```



Następnie napisano własny skrypt nawiązujący połączenie ze sceną oraz uruchamiający silnik przy lewym kole robota.

```

%% Start connection
vrep=remApi('remoteApi');
vrep.simxFinish(-1);

clientID = vrep.simxStart('127.0.0.1',19999,true,true,5000,5);

%% Main program

% left_Motor
[returnCode, left_Motor] =
vrep.simxGetObjectHandle(clientID,'bubbleRob_leftMotor',vrep.simx_opmode_blocking);

[returnCode] = vrep.simxSetJointTargetVelocity(clientID,left_Motor,2,vrep.simx_opmode_blocking);

[returnCode, detectionState, detectedPoint,~,~] = vrep.simxReadProximitySensor(clientID,
front_sensor, vrep.simx_opmode_streaming)
for i=1:50

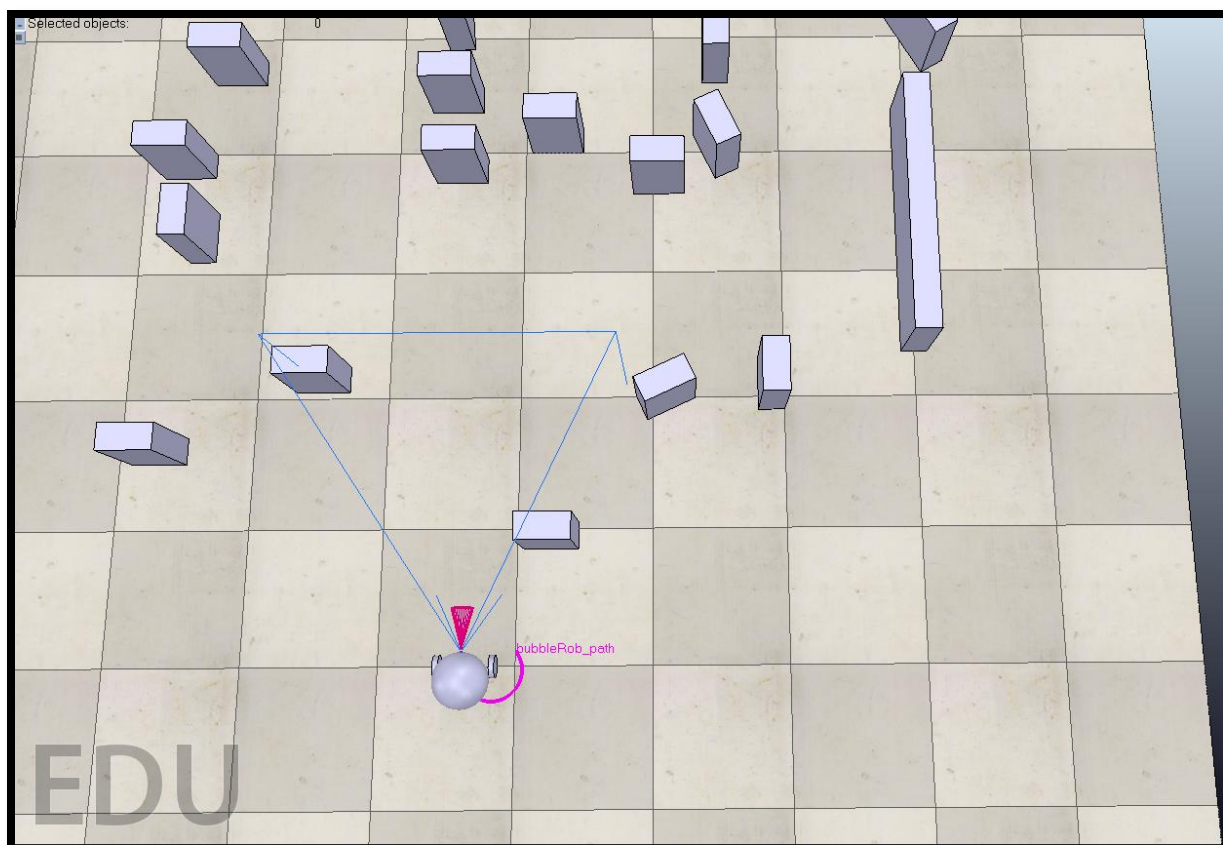
    pause(0.05)
end

[returnCode] = vrep.simxSetJointTargetVelocity(clientID,left_Motor,0,vrep.simx_opmode_blocking);
[returnCode] = vrep.simxSetJointTargetVelocity(clientID,right_Motor,0,vrep.simx_opmode_blocking);
%% Close connection
vrep.simxGetPingTime(clientID);

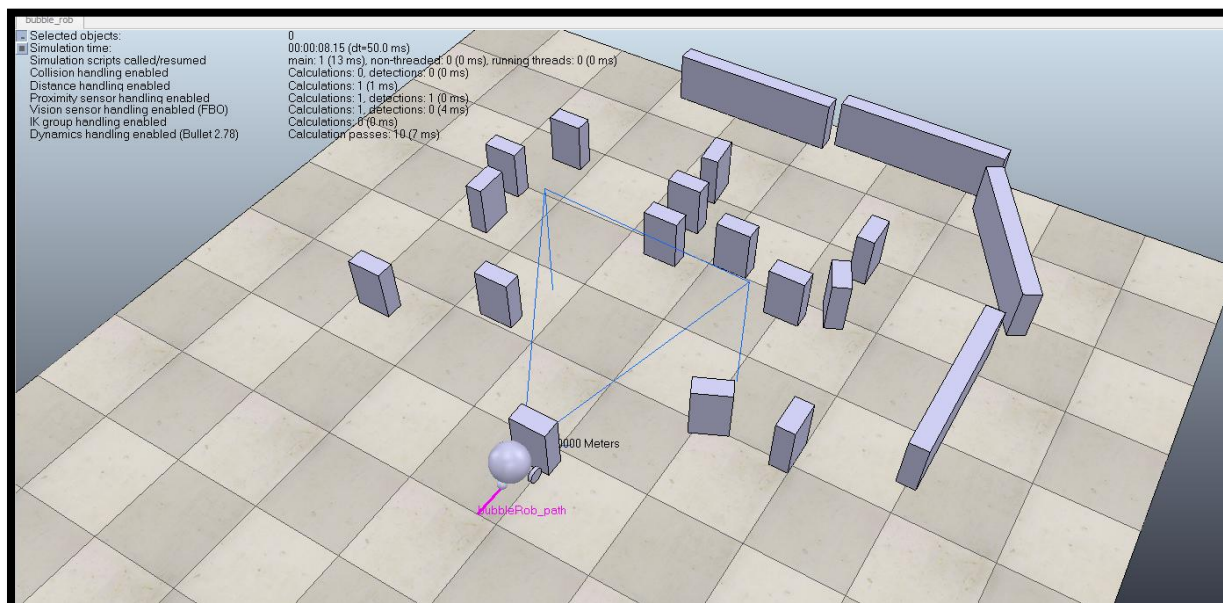
% Now close the connection to V-REP
vrep.simxFinish(clientID);
vrep.delete(); % call the destructor

```

Robot poprawnie przejechał chwilę po łuku, po czym zatrzymał się.



Następnie dodano uruchomienie również silnika przy prawym kole. Spowodowało to jazdę robota na wprost. Dodano również uchwyt czujnika odległości oraz odczyt jego wartości cyklicznie wykonywane w pętli for().



```
Command Window
New to MATLAB? See resources for Getting Started.

detectedPoint =

    1×3 single row vector

    0.0071    -0.0012    0.1000

returnCode =

    0

detectionState =

    uint8

    1

detectedPoint =

    1×3 single row vector

    0.0115    -0.0006    0.1000
```

Wykorzystany do tego kod:

```
%% Start connection
vrep=remApi('remoteApi');
vrep.simxFinish(-1);

clientId = vrep.simxStart('127.0.0.1',19999,true,true,5000,5);

%% Main program

% left_Motor
[returnCode, left_Motor] =
vrep.simxGetObjectHandle(clientId,'bubbleRob_leftMotor',vrep.simx_opmode_blocking);
% right_Motor
[returnCode, right_Motor] =
vrep.simxGetObjectHandle(clientId,'bubbleRob_rightMotor',vrep.simx_opmode_blocking);

[returnCode] = vrep.simxSetJointTargetVelocity(clientId,left_Motor,2,vrep.simx_opmode_blocking);
[returnCode] = vrep.simxSetJointTargetVelocity(clientId,right_Motor,2,vrep.simx_opmode_blocking);

% sensing_Nose
[returnCode, front_sensor] = vrep.simxGetObjectHandle(clientId,'bubbleRob_sensingNose',
vrep.simx_opmode_blocking);

[returnCode, detectionState, detectedPoint,~,~] = vrep.simxReadProximitySensor(clientId,
front_sensor, vrep.simx_opmode_streaming)
```

```

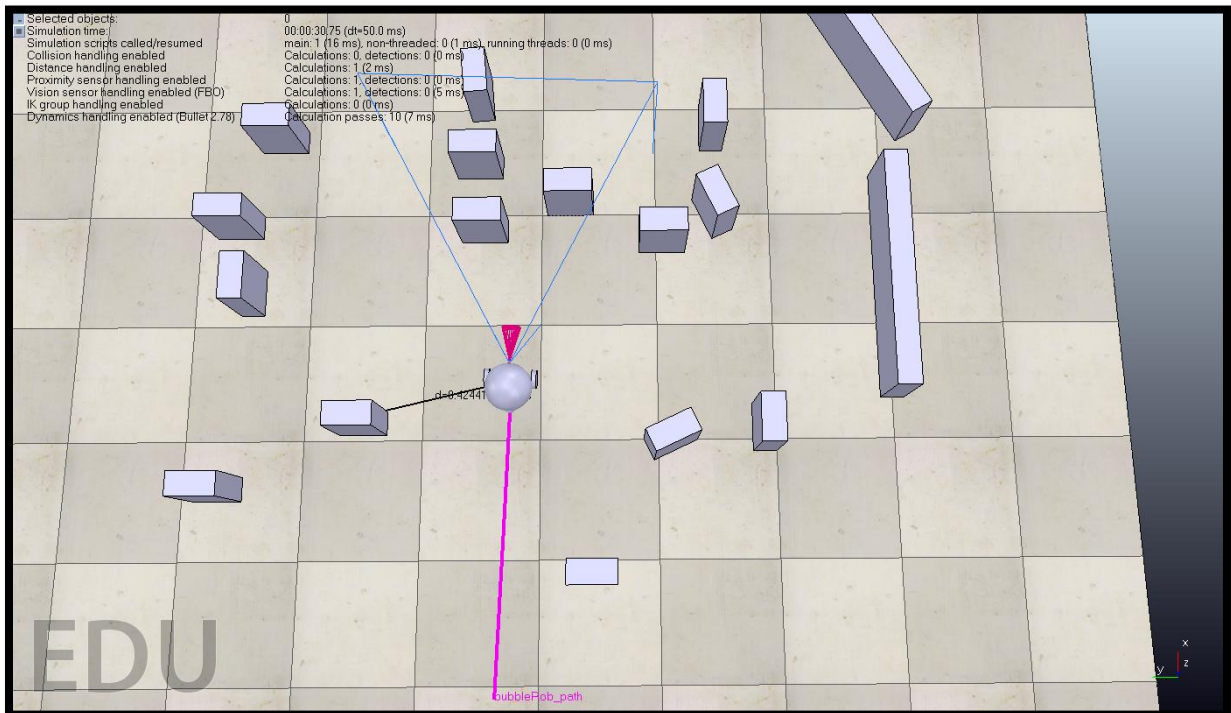
for i=1:50
    [returnCode, detectionState, detectedPoint,~,~] = vrep.simxReadProximitySensor(clientID,
    front_sensor, vrep.simx_opmode_buffer)
    pause(0.05)
end

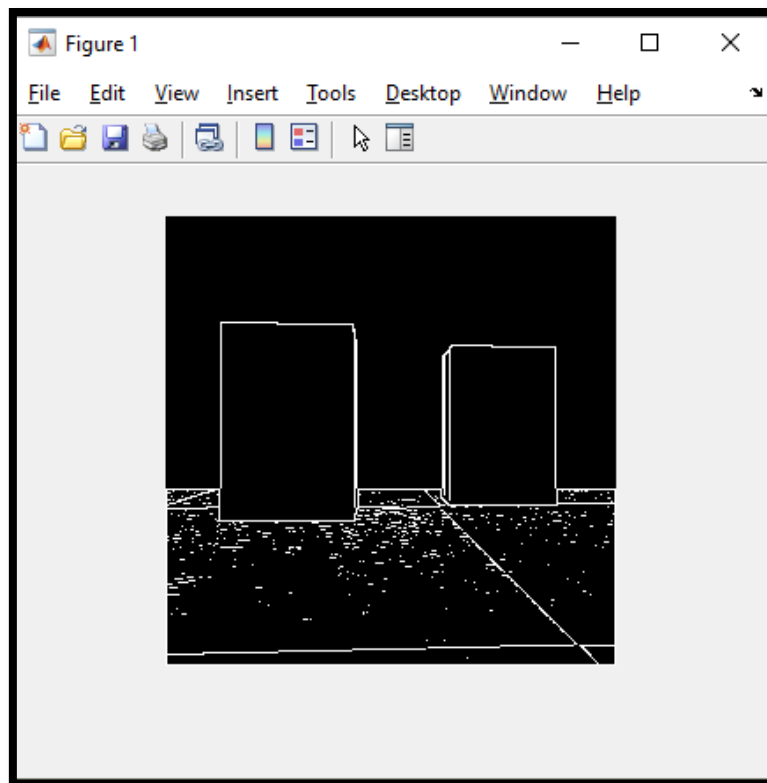
[returnCode] = vrep.simxSetJointTargetVelocity(clientID,left_Motor,0,vrep.simx_opmode_blocking);
[returnCode] = vrep.simxSetJointTargetVelocity(clientID,right_Motor,0,vrep.simx_opmode_blocking);
%% Close connection
vrep.simxGetPingTime(clientID);

% Now close the connection to V-REP
vrep.simxFinish(clientID);
vrep.delete(); % call the destructor

```

Na koniec dodano uchwyt czujnika wizji oraz w pętli for() cykliczne wywołanie funkcji *imshow(image)* które powoduje wyświetlenie widoku z czujnika robota.





Wykorzystany kod:

```
%% Start connection
vrep=remApi('remoteApi');
vrep.simxFinish(-1);

clientId = vrep.simxStart('127.0.0.1',19999,true,true,5000,5);

%% Main program

% vision sensor
[returnCode, sensorHandle]=vrep.simxGetObjectHandle(clientID,'Vision_sensor'
,vrep.simx_opmode_blocking)
[returnCode, resolution, image] = vrep.simxGetVisionSensorImage2(clientID,
sensorHandle,0,vrep.simx_opmode_streaming)

% left_Motor
[returnCode, left_Motor] =
vrep.simxGetObjectHandle(clientID,'bubbleRob_leftMotor',vrep.simx_opmode_blocking);
[returnCode, right_Motor] =
vrep.simxGetObjectHandle(clientID,'bubbleRob_rightMotor',vrep.simx_opmode_blocking);

[returnCode] = vrep.simxSetJointTargetVelocity(clientID,left_Motor,2,vrep.simx_opmode_blocking);
[returnCode] = vrep.simxSetJointTargetVelocity(clientID,right_Motor,2,vrep.simx_opmode_blocking);

% sensing_Nose
```

```

[returnCode, front_sensor] = vrep.simxGetObjectHandle(clientID, 'bubbleRob_sensingNose',
vrep.simx_opmode_blocking);

[returnCode, detectionState, detectedPoint,~,~] = vrep.simxReadProximitySensor(clientID,
front_sensor, vrep.simx_opmode_streaming)
for i=1:50
    [returnCode, detectionState, detectedPoint,~,~] = vrep.simxReadProximitySensor(clientID,
front_sensor, vrep.simx_opmode_buffer)
    imshow(image);
    [returnCode, resolution, image] =
vrep.simxGetVisionSensorImage2(clientID,sensorHandle,0,vrep.simx_opmode_buffer)
    pause(0.001)
end

[returnCode] = vrep.simxSetJointTargetVelocity(clientID,left_Motor,0,vrep.simx_opmode_blocking);
[returnCode] = vrep.simxSetJointTargetVelocity(clientID,right_Motor,0,vrep.simx_opmode_blocking);
%% Close connection
vrep.simxGetPingTime(clientID);

% Now close the connection to V-REP
vrep.simxFinish(clientID);
vrep.delete(); % call the destructor

```