VILNIUS UNIVERSITY

FACULTY OF MATHEMATICS AND INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE



DRAWING ROBOT "BOT'Vinčis"

Project work

Group members:

Artūr Chramčenko  
 Liudas Indrašius  
Žygimantas Ulevičius  
Domantas Varapnickas

Vilnius, 2014

# Contents

[Contents 2](#_Toc399978109)

[Terminology 3](#_Toc399978110)

[Summary 4](#_Toc399978111)

[Santrauka 5](#_Toc399978112)

[Introduction 6](#_Toc399978113)

[Main part 7](#_Toc399978114)

[I. Analysis 7](#_Toc399978115)

[II. Design 9](#_Toc399978116)

[III. Implementation 10](#_Toc399978117)

[IV. Testing 11](#_Toc399978118)

[References 12](#_Toc399978119)

[Anexes 13](#_Toc399978120)

# Terminology

leJOS - a [firmware](http://en.wikipedia.org/wiki/Firmware) replacement for [Lego Mindstorms](http://en.wikipedia.org/wiki/Lego_Mindstorms) programmable bricks. It includes a [Java virtual machine](http://en.wikipedia.org/wiki/Java_virtual_machine), which allows Lego Mindstorms robots to be programmed in the [Java programming language](http://en.wikipedia.org/wiki/Java_(programming_language)).

leJOS is a tiny Java Virtual Machine. leJOS NXJ includes all the classes in the [NXJ API](http://www.lejos.org/nxt/nxj/api/index.html) as well as the tools used to upload code to the NXT brick.

# Summary

# Santrauka

# Introduction

Today more and more robots are being used for daily human needs. It becomes crucial to learn to use robots to our advantage and expand their capabilities. Our project - drawing LEGO NXT 2.0 robot could be first step to create a device capable of drawing large scale paintings, schemas or other kind of works, which are too large for casual printers. The idea of drawing robot is not new and has already been implemented. Although, we are going to try to achieve similar results with more limited resoursers. // The goal of the project will be considered as achieved if we manege to create robot and its software capable of interprating countoured picture and drawing it on a paper.

## I. Analysis

At the beginning we did not have the robot, so the firs task was to get information and capabilities of it.   
  
From the whole list of parts, such as touch sensosr, three tacho motors, ultrasound distance sensor, light and color sensor, the main brick with CPU and hundreds of non-electronic parts several possible ideas were born:

* to make robot having only motors and orientating it by calculation of motor turnings;
* to add a light/color sensor and make robot follow projected lines on the paper.

The secont option after discussion appeared to be easier for simple, continious lines, but not sufficient enough to draw complex pictures, so we decided to use only the main brick, three motors, wheels and supporting part for the robot's chasis and marker lifting unit. That means that all calculation of robot's current position, distances, turn angles have to be calculated through motor turns. Although if minght not appear difficult task at first because of existing libraries, the robot has to make so many turns and moves, that we have to take margins of errors into serious consideration.

The motors are precise, can be easily programmed to function together in order to move forward or backward or to turn in oposite directions to turn robot around. They also have programable speed options.

The main NXT 2.0 brick is quite weak at first look. The main 32-bit microcontroller has 256 KB flash memory and 64 KB RAM. That means that the controller is not able to calculate difficult operations and has to receive only simplistic (?) commands. It has 6 input ports for sensors and three output ports for the motors. It has USB port, so the robot can be controlled via USB cable, or, in our case, use Bluetooth Class II V2.0 connection. The brick is powered by six AA rechargeable batteries;.

We had possibility to meet students who previously worked with the robot. They shared some basic information about firmware and programming of the robot (which is done by using Java language)

After the robot was finished we started gathering information about vectorizing images, extracting contour of an image.

After the research was complete, the list of probably useful list of libraries was made:

## II. Design

First time we saw the robot we are working with was the 24th of September. By the end of September we already had fully built and functioning robot. It was linked with computer via bluetooth and could be controlled by writing simple commands.

We have discussed several possible designs of our robot: make it with tracks, four wheels or three wheels. Because we needed high maneuverability and not the ability to drive over rough terain, we chose to make our robot with three wheels: two drive wheels and one small wheel for support.

Full list of parts used:

* two motors for each of main wheels so the robot could turn easily
* one motor for lifting marker up and down;
* a supporting wheel at the back of the robot : the main tasks for is to keep balance and robot horizontally, also reduce friction;
* main body of robot with the NXT brick on top;

One of the major problems we faced while disegning the robot was adjusting the marker so the end of it would be located precisely in the midle of the axis connecting drive wheels so it remained in place when the robot is turning. Schematics shown in picture...

## III. Implementation

At first we found libraries to communicate with the robot via bluetooth. Onother breakthrough was to be able to give simple commands to move forward:

to lift/put down marker:

draw circle:

and so on.

Image processing:

We have made GUI to make working with the pictures and robot easy:

1) User selects image he wants to be drawn (implemented file type restrictions so only image types can be selected)

2) Selected image is resized in proportion to be 2.97x4.20 by using thumbnailator-0.4.7-all.jar library;

For steps from 3 to 6 graphics2d was used:

3) Image is two times brightened in order to get more precise contours;

4) Selected image is showed on the left side of the window;

5) Image is converted to binary (black/white) image and showed on the right side of the window;

6) User can change the intensiveness of the image (amount of black lines (spots) to be drawn);

7) Final image is sent to processer class and special algorythm creates array of coordinates of black points (at first recognizes only horizontal and vertical lines);

8) Coordinates are saved in a output.txt file, nxjupload.bat program sends it to the NXT microcontroler together with launcher.nxj program which processes the output.txt file and begins the drawing process.

After that a new problem showed : the robot drew horizontal lines and after that vertical lines. This meant that not only many unnecessary moves were made, but precision of drawing was drastically lower. In order to solve this problem an algorythm was writen to sort the array of coordinates, so that the ending of one line was the begginig of another. By using this algorythm robot drew closed-contour figures in order.

## IV. Testing

# References

http://www.lejos.org/  
http://mindstorms.lego.com/  
http://www.robotnav.com/

# 

# Anexes

## Parts needed



2 x



2 x



3 x



2 x



3 x



6 x



3 x 5M



1 x 9M



3 x



1 x



1 x



1 x



1 x



1 x 12M



4 x 7M



4 x



6 x



4 x



2 x



1 x



1 x



12 x



1 x



2 x



1 x



2 x