Assignment 3

Cupak Miroslav,
David Nemecek,
Nizan Juraj,
Streck Adam

April 10, 2012

1 Concept

We want to build a robot capable of:

1. Calibrating its track and light detection sensor.
2. Reading instructions in some graphical represenatation while riding over them.
3. Visible execution of those instructions.

2 Design

2.1 Hardware

For self-calibration of the position is best to use three wheel vehicle with signe free wheel in the front and two wheels with independent motors in the back (sort of the thing that is built right now).
However, I think that 4-wheel vehicle with single motor for movement and second one for steering will be better for execution of instructions and overall more awesome, so I suggest building that one - there is a howto for one pretty, I would build on that: `http://www.nxtprograms.com/NXT2/race_car/steps.html` .

2.2 Input

We have two sensors so I would reccomend using two paralel lines of instructions that are printed on the paper - it is simple, transportable and we can make programs of any lenght using a glue ;).

2.3 Behaviour

There are four basic things such a robot could do:

1. Move and steer.
2. Make sounds.
3. Draw on LCD of the NTX brick.
4. Use third motor to do something.

I would pick two of them - move and make sounds, unless movement is too loud for sound to be audible, I will fruthemore explain how to utilize them.

3 Coding

3.1 Code representation

I see two ways how to make coding (I already nicknamed the language *GreyDot*, do you agree?):

- Use scale for gray with 10 steps.
- Use triples of black (0) and gray (1).

First option would be better, but we need to test if we can always calibrate the sensor so it reads correct instructions no matter what light conditions are - but it should work, when I tested it, it was pretty sharp. In this case we have 9 steps for different instructions and 1 for background.
Second option uses only three steps - black for 0, grey for 1 and white for background and I am pretty positive that this option would work even without self calibration. We would than have 8 combinations.

3.2 Semantics

I would use this two sets of instructions:

- Movement
  1. Move forward straight.
  2. Move forward left to make 30 turn.
  3. Move forward right to make 30 turn.
  4. Move backward straight.
  5. Move backward left to make 30 turn.
  6. Move backward right to make 30 turn.
  7. Stay / Stop.
- Sound
  1. Beep note A for 1/3 of period.
  2. Beep note A for 2/3 of period.
  3. Beep note A for 3/3 of period.
  4. Beep note B for 1/3 of period.
  5. Beep note B for 2/3 of period.
  6. Beep note B for 3/3 of period.
  7. Be silent.

One line of instructions would be for movement only, the other one would be for sound only.
After reading instructions our robot would always execute both instructions at once for a fixed period of time.

3.3 Writing instructions

I would write instructions in simple ASCII file and then have some script plot it.
I would recomment using two lines - first for movement with numbers of instructions of some symbolic representation - ¡ for steer left etc.

3.4 Example

Using only numbers of instructions following code:

```
222222
363636
```

would make a circling firetruck :)

3.5 Special symbols

There is at leas one symbol left for each line - these will be used as special symbols - for now we may need these symbols:

1. maybe - Start color calibration (should be black-black).
2. maybe - Start path calibration.
3. sure - Start reading instructions.
4. sure - End reading instructions and execute.

4 Calibration

We need to do following two calibrations, I am no sure in which ordering and if they start automatically or based on instructions.

4.1 Movement

We need to calibrate movement so when our robot starts reading instructions it goes straight - this must be precise.
For this we will need at least two-papers long lines.

4.2 Light

To assure correctness of reading we need to set low and high value of the light input - there must be at least a bit of white and bit of black space on the paper.

4.3 Errors

It still might happen that our robot goes out of the line before the end - I suggest to use timeout - if there is no new instruction on the input within given time, robot stops.

5 Code

5.1 Architecture

Calibration, reading and execution are independent parts so I would keep them that way in the code - they can be even created independently, which seems like an OK idea to me.
On the other hand I would suggest using common conventions for syntax.