### Tags: Platform

*DisCoS (Distributed Control System) is a control network framework for any collection of robotic devices. Its innovation lies in the concept of sending missions rather than commands. This is possible because of the .bot programming language we created. Using DisCoS, we created the TNT education and robotics platform, which allows users to write live code and programs for NXTs. These are sent to android devices as .bot packages, which are then compiled on the fly and used to control NXT rovers.*

This project is solving the [**Lego Rovers**](https://2013.spaceappschallenge.org/challenge/lego-rovers) challenge.

**Description**

**The App**

We've built the TNT **learning and robotics platform** as a first implementation of DisCoS.

With only a browser, connect to our **Lego NXT rovers** and solve challenges by writing live code. We've developed the TNT (Turtle Not Turtle) programming language, which is optimized for rapidly developing powerful robotics programs.

TNT is a full DisCoS system where the **user serves as the Master** (controlling many rovers at a time),**Android devices serve as Controllers** (talking to the Master via. wifi and the Platforms via. bluetooth), and the **NXT rovers serve as the Platforms**. Code is compiled from TNT into .bot on browser, and sent as a JSON package to the Android device, which executes the code in a Java virtual machine and sends relevant commands to the NXT rover.

**The Idea: DisCoS**

 The DisCoS framework aims to create more powerful **collective robotics systems** by distributing computation and autonomy throughout the network. In a traditional master/slave system, the master micromanages every device. In DisCoS, the Master instead dynamically generates full programs, which are computed throughout the network, leaving micromanagement, and device-specific interfaces to the Controller. This allows the Master to focus solely on high level missions. The end result is a highly optimized, flexible and powerful **collective intelligence network**.

DisCoS devices operate at one of three levels. At the top is the **Master (i.e. a server)**, which has complete information of the network and as well as any relevant information from the individual devices in the network. Underneath are the **Controllers (i.e. an Android device)**, which execute programs run by the master which execute programs run by the master. At the bottom are **Platforms (i.e. a rover)**, the workers in the system that actuate DisCoS missions.

The Master receives only relevant and **necessary information** from the controllers throughout the network. It makes decisions on how the network should respond, and sends these throughout the network as missions. These missions can be complex, such as a full maze solver. Controllers receive these missions as .bot programs and see them through to completion by **micromanaging platforms**. Most information (i.e. sensor data) required to complete these missions are sent only between the Controller and Platform, and only required information is sent back up to the Master. The **Platforms** interfaces with the Controller through a device specific driver, obeying **direct commands** from the Controller.

**Core Concepts:**

**- Distributed autonomy:** the master sends missions, not commands. The master is free to operate only at high levels, leaving controllers to micromanage platforms and to complete individual missions.  **- Generality:** DisCoS can be implemented with any system of networked devices. Because .bot implements only abstracted high level commands, everything from lightbulbs to rovers can be networked together.  **-RESTful:** No proprietary network protocols or encoding. The entire DisCoS system communicates through HTTP get and post requests, and all messages and .bot programs are JSON objects.

**The Method: .bot**

 .bot is a turing complete, functional programming language we created in order to make DisCoS possible. **Expressed entirely in JSON,** .bot allows entire programs to be sent using **RESTful APIs** on which DisCoS is built. We created a virtual machine to interpret .bot code, written in Java - allowing any Java capable device (including all android devices) to be able to execute .bot code and act as a controller in the DisCoS system.

.bot code includes only high level functions that are consistent across devices (such as move forward, check distance in front, etc). The controller simply pings the master to let it know which functions are available, and the master can then **write entire programs** for the platform without needing any knowledge of the actual implementation. The controller contains a driver that implements low level hardware functions (such as move forward). This allows code to be run virtually, and thus, bad code will **never crash** any software. Code can be paused, stopped, and reflashed all on the fly.

**Core Concepts:**

**-Turing Complete:** .bot is a full turing complete, interpreted, functional programming language. It has complete support for function definitions, recursion, loops, conditionals, common mathematical operations. Functions are first-class citizens. **-Virtual Machine:** We wrote a virtual machine in Java to compile .bot code. This allows it to run on its own thread on any Java capable device (such as Androids), without the risk of crashing the software on the device in the event of bad code. Also, code can be paused, stopped and reflashed instantly.  **-JSON:** All .bot programs can be expressed entirely in Javascript Object Notation. This allows programs to be compact, readable, and easily sent using only RESTful HTTP protocols on which DisCoS is built.

**The Implementation: TNT Learning Platform**

**TNT**, which stands for "TNT Not Turtle", is an educational learning and robotics platform. We created the TNT programming language, which is inspired by simply python/ruby syntax and the philosophy of the [Logo (Turtle)](http://el.media.mit.edu/logo-foundation/logo/) class of languages, which were created at MIT in the 1970s. We built an **IDE and compiler** for TNT in javascript, and it will run in any modern web browser. Hence, our audience is as broad as possible, as anyone with a web browser can utilize the TNT learning platform.

On the TNT platform, the user/student serves as the Master in a DisCoS system. We networked several android devices as Controllers, which constantly communicate with our web server. Each Android is paired via bluetooth to a Lego NXT Rover. The user is presented with **several challenges** (solving mazes / puzzles), and they are challenged with writing programs for the NXT Rovers, **controlling one or several** at a time, in order to solve these challenges. They can also write step by step commands (such as forward(2) ), which will be **immediately executed** by the rover. We believe that this is an effective educational approach to engage students and get them excited about computer science / engineering / robotics, as they can see a rover moving in front of them. Our motto is **"push robots not pixels"**.

Simple TNT programs are compiled in the browser to **.bot JSON objects**, which are pushed to our server and sent to the relevant Controllers. Because .bot is interpreted in a **virtual machine**, there is no worry about bad code crashing the software, nor does the student need to worry about compiling / drivers. In fact, **the student need not even own an actual NXT device** (which can be pricey), as they can watch their rover perform challenges over a live webcam feed. Thus, anyone with a web browser is able to jump on the TNT learning platform and **start writing code instantly**.

**Core Concepts:**

**-Push Robots not Pixels:** See results as you type. A simple fd(2) command will send the rover moving forward for 2 seconds. **-TNT Not Turtle:** Based off MIT Turtle and inspired by python / ruby, TNT strips away complexities and let's you focus on logic whilst still being a complete programming language. TNT supports variables, threading, functions, loops, and conditionals. **-Write anywhere:** We've built both the the compiler and IDE (with syntax checking and indentation) in javascript. Therefore, all the code can be written and compiled from any modern browser. No downloads or expensive equipment are required to use TNT!

**Credits**

DisCoS was created by team hackAD, a group of students studying at New York University Abu Dhabi.

Team Members & Countries of Origin:

* **Lingliang Zhang** - New Zealand
* **Moïri Gamboni** - Australia
* **Oleg Grishin** - Russia
* **Rodrigo Ceballos** - Colombia
* **Mariko Kuroda** – Japan
* **Soichiro Hattori** - Japan

**Project Information**

* **License:** [GNU General Public License](http://opensource.org/licenses/gpl-license)
* **Source Code/Project URL:** <https://github.com/grishinoleg/hack_ad>

**Resources**

* **Project Website** - <http://discos.herokuapp.com/>
* **Live Web Demo** - <http://discos.herokuapp.com/use/robo>
* **Demo Video** - <http://www.youtube.com/watch?v=xKxGScd1rq8>
* **Extended Tech Explanation Video (6 mins)** - <http://www.youtube.com/watch?v=muxXGI9fNek&feature=youtu.be>