Arduino Robotics Camp

v0.9
16 February 2016

Designed and written by Kristof Aldenderfer and Luca Angeleri

Introduction

Welcome to Arduino Robotics camp, instructor! Over the next five days, your students will be designing, constructing, testing, assembling, and playing with their very own autonomously- and directly-controlled tracked robots, which they'll be controlling from a mobile device. This involves the laser cutters, 3D printers, Arduino IDE, and lots of wiring.

Throughout this project, the code base should match the wiring of the schematics. If you prefer to wire the system differently, just make sure to adjust the code accordingly.

Required hardware (provided by TechShop):

- pjrc's Teensy 3.2
- Adafruit BlueFruit LE UART Friend
- Adafruit TB6612 Dual Motor Driver
- Lithium polymer battery (500mAh+)
- Tamiya twin motor gearbox
- Tamiya track and wheel set
- Breadboard and jumper kit
- Components from Arduino Wiring (formerly EEE205: Arduino Part 2)
- Laptop or desktop computer

Additional materials (provided by TechShop):

- 24” x 18” x 1/8” acrylic or wood sheet (one sheet per eight students)
- Acrylic cement
- Mounting screws and nuts

Required hardware (provided by student):

- Mobile device with Bluetooth 4.0 capability

Required software and documents:

- Arduino IDE (1.6.5+)
- Teensyduino
- Adafruit Bluefruit LE Connect (Android or iOS, depending on student's device)
- TS Robotics Camp Github repo

Daily Schedule Summary

This schedule is subject to change based on instructor's analysis and discretion. There is time on both Thursday and Friday for advanced topics if the students are ahead of schedule, or for brevity of topic if the students are behind schedule.

Monday, day 1: A primer on electronics

Introduction
Chat about general electronics as well as robotics
Play with littleBits
Install software (computer and mobile device)
Go through MC1 (formerly EEE205: Arduino Part 2)
For tomorrow: discuss DC motors

Tuesday, day 2: Make some movement!

Gearboxes
Solder motor and motor driver connections
Explain and connect motor driver to MC and gearbox
Code the gearbox
"Drive" the breadboard
For tomorrow: discuss mechanical engineering and chassis etching

Wednesday, day 3: Industrial design and fabrication

Abbreviated laser cutter course
Illustrator tutorial
Design the etched parts of the chassis
Laser cut the chassis
Cement the chassis (with gearbox installed)
Assemble the track and wheel set
Test the tank
For tomorrow: discuss Bluetooth and autonomy

Thursday, day 4: Programming Bluetooth and autonomous motion

Discuss Bluetooth connectivity
Connect and begin coding Bluetooth
For tomorrow: finally bringing it all together, and challenges

Friday, day 5: Final tweaks and tank challenges

Code cleanup and tweaks, plus Github
Wrap up any final details
Design and build challenge course
Run the challenge course
Show and tell!

Daily Schedule Expanded

Monday, day 1: A primer on electronics

Overview

Today is all about getting an understanding of basic electronics and microcontroller. Ultimately, students will have the Teensy 3.2 microcontroller up and running, and play with its inputs and outputs via Arduino's Example sketches - effectively an extended Arduino Wiring course, with a different microcontroller.

Hardware introduced today:

- teensy 3.2
- breadboard and jumper kit
- components for Arduino Wiring course

Introduction

Introduce yourself as well as the camp. Outline what they will be doing during the week. Ask the students if they have experience with electronics or robotics. Hand out hardware.

Play with littleBits

The basic kit will suffice. This will likely be a review for older students, but completely new for younger students. Be sure to distinguish among power, input, and output.

Install software (computer and mobile device)

Each student will need to have a computer with the following installed:

- Arduino IDE version 1.6.5+
- Teensyduino

as well as a Bluetooth 4.0 capable device. The Arduino IDE is for writing code for the Teensy 3.2; Teensyduino is for programming the actual hardware. You may substitute microcontroller; the code will work on any Arduino-compatible hardware. Only the pin assignments will need to be adjusted to the particular hardware.

Go through Arduino Wiring (formerly EEE205: Arduino Part 2)

Connect the Teensy 3.2 to the computer and make sure the IDE will upload the Blink sketch to it. The process is almost identical to a standard Arduino board, except that once the IDE is done compiling, Teensyduino will launch. If Teensyduino is in "Auto" mode, the code will immediately be uploaded, and the software will say "Reboot OK". If Teensyduino is in manual mode, the "Reset" button must be pressed on the Teensy 3.2.

From here, Arduino Wiring can be taught, paying particular attention to analog inputs and outputs. If students are advanced, there should be time to let them experiment with the codebases.

For tomorrow: discuss DC motors

Note that Arduino Wiring does not address motor control. Have the students chat about controlling motors directly with the Teensy 3.2; is this even possible?

Tuesday, day 2: Make some movement!

Overview

Today is all about building and integrating the gearbox into the MC environment. Students will assemble their gearboxes, and then connect and drive them via the Teensy 3.2. There is quite a bit of conceptual and contextual material to do today.

Hardware introduced today:

- Gearbox
- Motor driver

Gearboxes

Tanks have no steering column, as cars do; so how do they move and turn? Introduce the gearbox, and gearing in general. Instructions on how to assemble the gearboxes are given with the kit. They can be assembled in three different configurations; the “C type” is the current ideal configuration, as it gives the highest torque. This section can be frustrating to students, as the parts are small, and require slow and careful reading and assembly.

Solder motor and motor driver connections

(If students have never soldered before, give a brief safety overview and a tutorial on how to solder.) The leads from the motors should be approximately 15cm long, and should be left unconnected on the opposite end. Power leads should be soldered directly from the motor driver’s power pins; they should be approximately 5cm in length, and also left unconnected on the opposite end. The male header pins can be soldered directly to the motor driver.

Explain and connect motor driver to MC and gearbox

How do you get a motor to spin? How do you get it to then spin in the opposite direction? Get students to surmise why a motor driver is required. Hook up the motor drivers as shown below:

INSERT PICTURE

Code the gearbox

Ask students to conceptualize how to spin the motors in an arbitrary direction. The goal is to have them code four functions: `forward()`, `backward()`, `left()`, and `right()`, and use those functions to write a “directionDemo” sketch, which loops a fixed command set. See github repo for full code. Instructor is free to provide as little or as much of the sample code to students, dependent upon their coding skill level.

“Drive” the breadboard

Gently secure the gearbox to the bottom of the breadboard. Take out a pair of drive wheels from the track and wheel set; the diameter does not matter. Attach them to the gearbox axles, and have the students upload their “directionDemo” code. Experiment.

For tomorrow: discuss mechanical engineering and chassis etching

As students how they will turn this setup into an actual driving system. Introduce mechanical engineering as a concept, and talk about fabrication. Have students begin to think about how they want their chassis decorated.

Wednesday, day 3: Industrial design and fabrication

Overview

Today students will learn the basics of CAD and design the rastered portion of the chassis, then actually go through the fabrication process, and assemble the chassis with the gearbox mounted.

Hardware introduced today:

- Laser cutter
- Track and wheel set

Abbreviated laser cutter course

Teach an abbreviated laser cutter class, so students can contextualize what they'll be doing today.

Illustrator tutorial

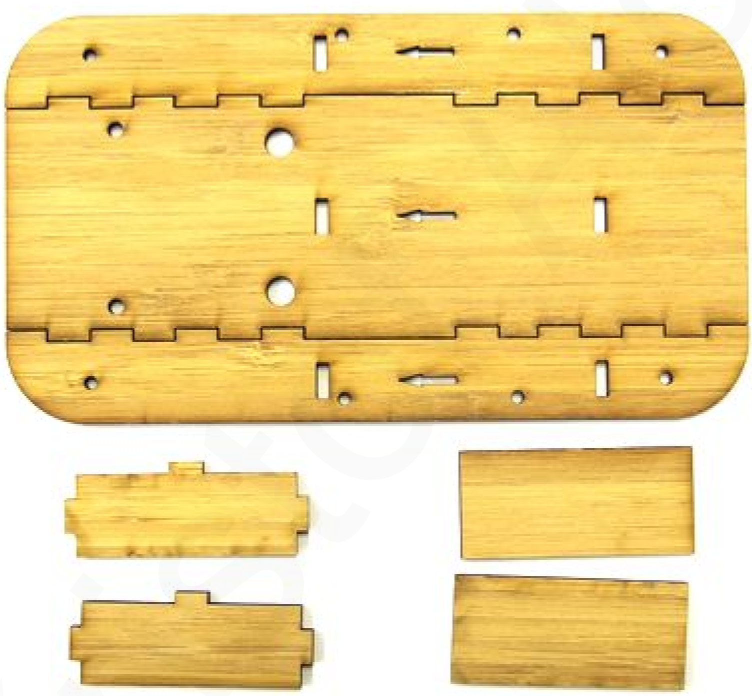
Give a simple vectoring and rastering lesson.

Design the etched parts of the chassis

Students can then begin to design their chassis based on the template called "Chassis Design" - this is found in the Github repo under "Support Files". Make sure students do not add any stray vectors which would cut through the chassis.

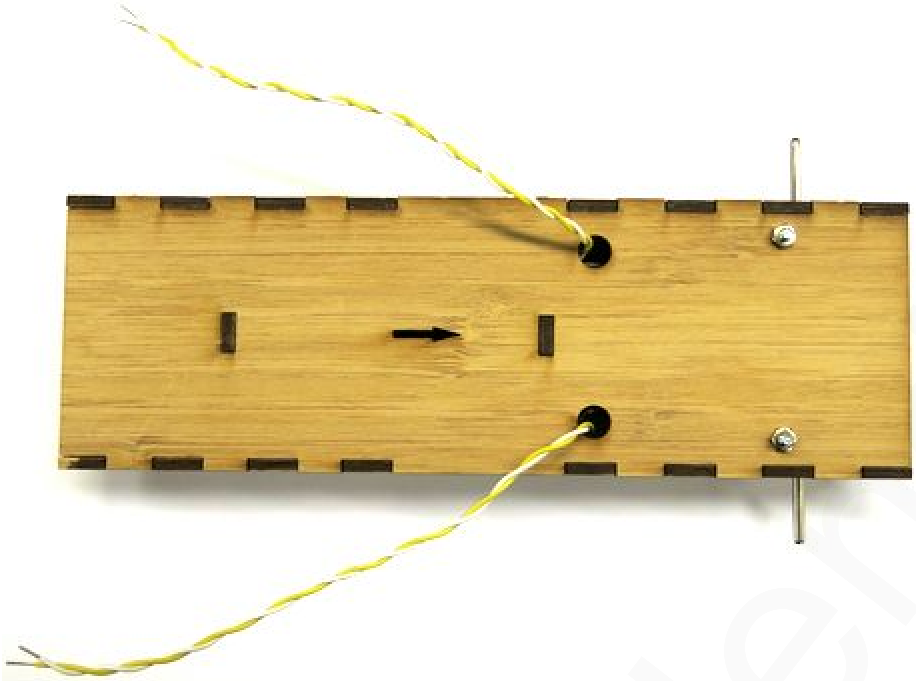
Laser cut the chassis

You may prefer to combine students' designs into one document to save time. Up to 8 chassis can fit on one 24" by 18" sheet of material.



Cement the chassis (with gearbox installed)

The chassis is designed with the gearbox permanently installed. To do this, first disconnect the motor leads from the breadboard. Start assembling the chassis by cementing one of the long runners to the rectangular base, then cement the lower upright supports; run the motor leads through the chassis base holes, and insert the gearbox into the chassis; then mount the second long runner. For best adhesion, the cement must set for several hours.

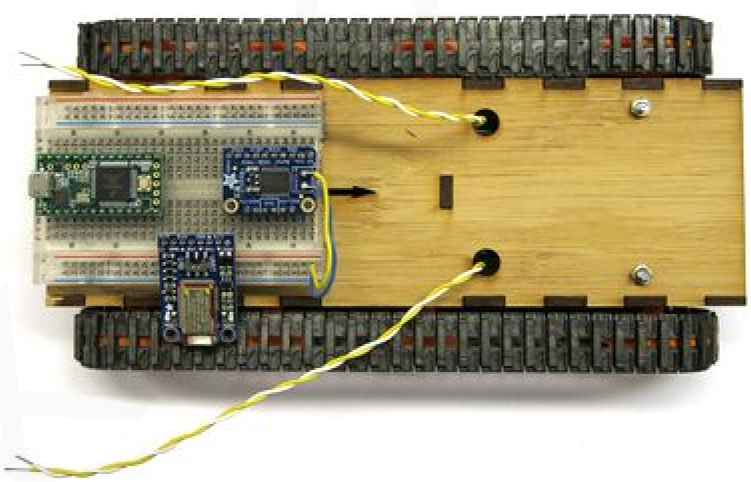


Assemble the track and wheel set

The tank uses the following from the track and wheel set:

- 1 long, 2 medium, and 2 short track sections (per track)
- 2 large drive wheels (or "sprockets")
- 2 large idler wheels
- 6 medium idler wheels
- 3 round idler axles
- 8 end caps

Additionally, the caps on the end of the drive wheels will need to be clipped, and the plastic on each idler wheel which covers the axle will need to be snipped ever so slightly, so that they will fit on the chassis properly.



Test the tank

Though the cement should set for several hours, students can wire the gearbox back into the breadboard and see their tank move for the first time.

For tomorrow: discuss Bluetooth and autonomy

Now that the chassis is fully built and the basic electronics are in place, discuss how to improve upon the code base. How could they communicate with it remotely? What other modes might they want their tank to operate in?

Thursday, day 4: Programming Bluetooth and autonomous motion

Overview

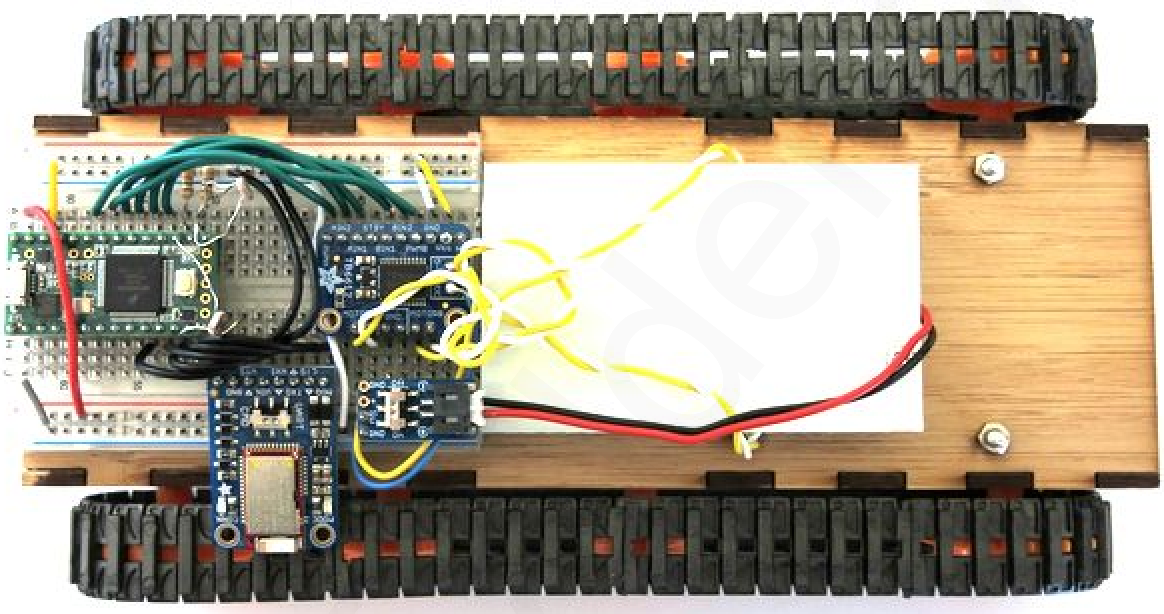
Students' tanks currently only operate in a "direction" demo mode, which shows simple, repetitive movement. Today they'll be adding manual motion

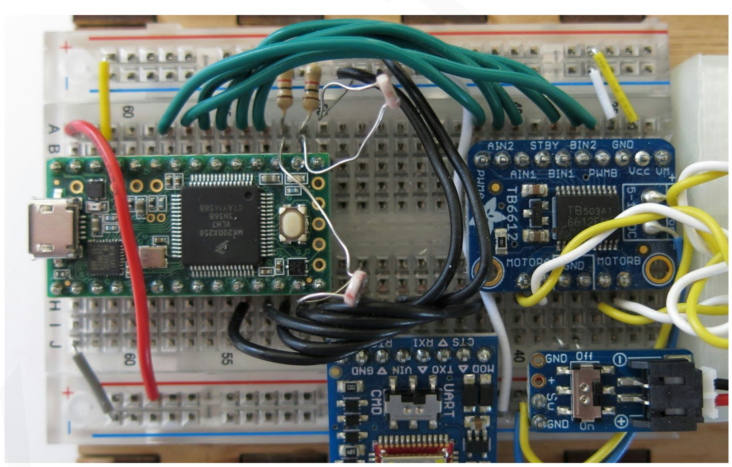
Hardware introduced today:

- adafruit Bluefruit LE UART friend

Discuss Bluetooth connectivity

Connect and begin coding Bluetooth





For tomorrow: finally bringing it all together, and challenges

Friday, day 5: Final tweaks and tank challenges

Overview

The last day of camp sees students doing final tweaks on their tanks, and then putting them to a test!

Hardware introduced today:

- none

Code cleanup and tweaks, plus Github

Make sure the students' codebases are as bug-free as possible. Have them comment generously so that they understand what they've done, and can reference it in the future. Have them create a github account (if they don't have one already) and push their very own new repo.

Wrap up any final details

Time has been allotted for catch up, in case students are behind schedule. If students are on schedule, they can design and add a fourth mode to their tanks. (Make sure they push to the repo again afterward.) If they are ahead of schedule, an abbreviated course on the 3D printers can be taught, so that students can design and print accessories for their tanks.

Design and build challenge course

Have students design and build an obstacle course for their tanks to manually navigate through. Assuming the gearboxes have been build in the "C-type" configuration, the tanks can climb over surprisingly large obstacles, and traverse inclines at upwards of 45° (if the incline has a sufficient coefficient of friction).

Run the challenge course

If students want to compete, have them each run the obstacle course individually for time.

Show and tell!

Show off the tracked robots to the rest of TechShop, and to the students' parents! Have them explain to their parents how the system is connected together, and what they learned.