## Contents

1 Introduction  
1.1 What is Tote? ................................................. 3  
1.2 How is it different from all those toys? ................. 4  
1.3 Challenges .................................................... 4  
1.4 Required Skills ............................................ 5  

2 Bill of Materials  
2.1 Required ....................................................... 7  
2.2 Optional ....................................................... 7  
2.3 Printed Circuit Board .................................... 8  
2.4 Arduino Pro Mini ........................................ 8  
2.5 SG90 Micro Servos ..................................... 9  
2.6 Pin Headers .................................................. 10  
2.7 1S LiPo Battery ........................................... 11  

3 Required Tools .................................................. 13  

4 Assembly Instructions for PCB Version 2  
4.1 Electronics ................................................... 16  
4.2 Programing .................................................... 33  
4.3 Mechanics ....................................................... 36  
4.4 Adjustments .................................................... 51  

5 Assembly Instructions for PCB Version 4  
5.1 Decide on the Version ....................................... 53  
5.2 The Servo Headers ....................................... 55  
5.3 Battery Header and Optional Jumper Headers .......... 58  
5.4 Optional Microcontroller Socket ......................... 62  
5.5 Alternative, Solder the Microcontroller Directly ..... 64  
5.6 Optional Battery Monitoring Circuit .................... 68  
5.7 Optional Anti-brownout Capacitor ....................... 72  
5.8 Power Switch ................................................ 74  
5.9 Optional IR Receiver ....................................... 75  
5.10 Optional Piezo Buzzer ................................... 82  
5.11 Servo Horns ............................................... 84  
5.12 Legs .......................................................... 89
## Assembly Instructions for PCB Version 5

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Solder the Microcontroller Board Directly</td>
<td>102</td>
</tr>
<tr>
<td>6.2 The Servo Headers</td>
<td>104</td>
</tr>
<tr>
<td>6.3 Infra-red Sensor</td>
<td>108</td>
</tr>
<tr>
<td>6.4 Power Switch</td>
<td>110</td>
</tr>
<tr>
<td>6.5 Battery Connector</td>
<td>111</td>
</tr>
<tr>
<td>6.6 Optional Piezo Buzzer</td>
<td>114</td>
</tr>
<tr>
<td>6.7 Optional Battery Monitoring Circuit</td>
<td>115</td>
</tr>
<tr>
<td>6.8 Optional Anti-brownout Capacitor</td>
<td>119</td>
</tr>
<tr>
<td>6.9 Servo Horns</td>
<td>121</td>
</tr>
<tr>
<td>6.10 Programming</td>
<td>123</td>
</tr>
<tr>
<td>6.11 Legs</td>
<td>125</td>
</tr>
</tbody>
</table>

## Electronics

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 Software</td>
<td>133</td>
</tr>
<tr>
<td>8.1 Features</td>
<td>133</td>
</tr>
</tbody>
</table>

## Modifying and Extending

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1 GPIO Pins</td>
<td>135</td>
</tr>
<tr>
<td>9.2 Sensors</td>
<td>135</td>
</tr>
<tr>
<td>9.3 Peripherals</td>
<td>135</td>
</tr>
<tr>
<td>9.4 Remote Control</td>
<td>136</td>
</tr>
<tr>
<td>9.5 Additional Boards</td>
<td>136</td>
</tr>
</tbody>
</table>

## Design Decisions

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1 Design Goals and Constraints</td>
<td>141</td>
</tr>
<tr>
<td>10.2 Hacks and Workarounds</td>
<td>141</td>
</tr>
</tbody>
</table>

## Inverse Kinematics

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1 Basic Trigonometry</td>
<td>144</td>
</tr>
<tr>
<td>11.2 Solving a Leg</td>
<td>144</td>
</tr>
<tr>
<td>11.3 Trim and Reverse</td>
<td>145</td>
</tr>
</tbody>
</table>

## Walking Algorithms

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1 Statically and Dynamically Stable Gaits</td>
<td>147</td>
</tr>
<tr>
<td>12.2 Area of Support</td>
<td>148</td>
</tr>
<tr>
<td>12.3 Moving Forward</td>
<td>150</td>
</tr>
<tr>
<td>12.4 The Creep Gait</td>
<td>151</td>
</tr>
<tr>
<td>12.5 The Trot Gait</td>
<td>152</td>
</tr>
</tbody>
</table>

## Step-by-step Tutorial

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.1 Soldering the Pin Headers</td>
<td>153</td>
</tr>
<tr>
<td>13.2 Connecting the Pro Mini to the Computer</td>
<td>155</td>
</tr>
<tr>
<td>13.3 Uploading First Arduino Program</td>
<td>156</td>
</tr>
<tr>
<td>13.4 Blinking a LED from the Computer</td>
<td>158</td>
</tr>
<tr>
<td>13.5 Attaching a Buzzer and Making Sounds</td>
<td>158</td>
</tr>
<tr>
<td>13.6 Attaching an RC Servo</td>
<td>158</td>
</tr>
<tr>
<td>13.7 Controlling the Servo and Buzzer from the Computer</td>
<td>158</td>
</tr>
<tr>
<td>13.8 Attaching an Infra-Red Sensor</td>
<td>158</td>
</tr>
<tr>
<td>13.9 Blinking a LED, Making Sounds and Moving Servos with a TV Remote</td>
<td>158</td>
</tr>
<tr>
<td>13.10 Making a Pan-and-tilt Gimbal</td>
<td>158</td>
</tr>
<tr>
<td>13.11 Attaching a Battery</td>
<td>158</td>
</tr>
<tr>
<td>13.12 Assembling the Printed Circuit Board</td>
<td>158</td>
</tr>
<tr>
<td>13.13 Monitoring Battery Voltage</td>
<td>158</td>
</tr>
</tbody>
</table>
13.14 Quadruped with Two Degrees of Freedom per Leg ............................. 158
13.15 Leg Inverse Kinematics ................................................................. 158
13.16 Unbalanced Crawling ...................................................................... 158
13.17 Crawling Sideways .......................................................................... 158
13.18 Rotating in Place ............................................................................. 158
13.19 Turning While Crawling ................................................................... 158
13.20 Changing Body Height ..................................................................... 158
13.21 Shifting the Body ............................................................................ 158
13.22 Balanced Crawling .......................................................................... 158
13.23 Trotting ......................................................................................... 158
13.24 Attaching a Proximity Sensor ............................................................. 158
13.25 Following an Object ......................................................................... 158
13.26 Avoiding Obstacles ......................................................................... 158
13.27 Attaching a Bluetooth/WiFi Module .................................................... 158
13.28 Telemetry ...................................................................................... 158
13.29 Attaching an Accelerometer ............................................................... 158

14 Feedback and Help ............................................................................. 159
14.1 Feedback .......................................................................................... 159
14.2 Help and Support .............................................................................. 159
14.3 Contact ............................................................................................. 159
Tote is a small (fits inside your palm), four-legged, walking robot, with three degrees of freedom per leg, Arduino for its brains and controlled either with a TV remote, or by additional electronics added on top of it. It is very simple, cheap and sturdy, for this class of robots. Its goal is to be a starting point for anyone who wants to start building multi-legged robots.

Contents:
What is Tote?

It’s a small walking robot, designed to be as cheap and simple to build as possible, while at the same time serving as a starting point for more complex projects and for teaching. In its basic form it just walks around and turns, controlled by a TV remote, but it can be easily extended by adding sensors and more interesting behaviors.
How is it different from all those toys?

Most walking toys available currently are not very different from the wind-up toys that our parents used to play with – except they have an electric motor instead of a spring. But the whole walking pattern is hard-wired in the mechanism of the toy, and it can do nothing more than walking forward and backwards on flat surfaces, and maybe turning.

This robot is potentially more than a toy. While the basic functionality that you begin with is pretty much the same, this robot can move each of the legs individually within three degrees of freedom, controlled with a computer program that you can freely modify. You can make it dance, you can make it climb, you can make it follow any movement plan. Furthermore, you can add sensors and modules to it, that would allow you to make it do practically anything – as long as you have enough skill to actually program it. Don’t get me wrong, it’s very hard to make it do even the simplest things – but the point is that you can!

Challenges

A lot of walking robot projects never get very far, because they turn out to be much more complex, expensive and fragile than initially anticipated. By making certain choices about size, materials and software, I managed to keep Tote relatively simple, affordable and robust. Thanks to its small size it has more than enough power to carry its weight, and the simple plastic parts are enough to hold it together and even survive falls from heights with just minor damage. At the same time, being just a starting point, Tote gives you a lot of room for improvement and experimenting.
**Required Skills**

First and foremost, you will need patience. A lot of decisions were made here trading time for money. You will wait for the printed circuit board to be made for you in a factory and then shipped to you. You will wait for shipment of all the parts you order online. Once it all arrives, you will spend hours debugging and fine-tuning the code. So this is not a project for aggressive deadlines.

You will also need to glue things together and assemble them using screws. You will probably also need to use a file to adjust parts that don’t quite fit. You won’t need a mechanical saw or drill, though (assuming you use a PCB they made for you in a factory – if you are hardcore, you can make the PCB yourself).

This project requires some basic soldering – at the level of soldering pin headers to a module, or soldering wires to a PCB. Be careful not to burn yourself.

You will have to properly connect the USB2TTL converter to your robot to program it from your computer using the Arduino IDE. For basics you don’t need to know how to code, as all the code is written for you, but you will probably want to modify and extend it once you get it to work.

Depending on how proficient you are in such things, it can take you anywhere from four to eight hours to get it all working.

In order to take the project further, you will need at least basic knowledge of Arduino programming and information about how the sensors that you want to use work, and how to manipulate the data that you receive from them. If you add more modules, you will also need to know how to program those. This is a huge topic, and you can keep learning and experimenting for years.
With the exception of the printed circuit board, which you have to make or order according to the provided designs, all parts are off-the-shelf, ready to use components which you can find in hobby stores, or even better, order online.

**Required**

- Printed Circuit Board, the board design file is in the repository,
- Arduino Pro Mini, 3.3V/8Mhz version,
- 12 SG90 Micro Servos, with screws and plastic horns,
- 2.54mm pin headers, with at least 38 pins total,
- 3.7V 1S LiPo Battery (I used a 300mAh battery),

**Optional**

- Infra-Red Receiver (I used VS1838B).
- Ultra Sound distance sensor (I used HC-SR04),
- 1000μF electrolytic capacitor, for avoiding brown-outs,
- 2 47kΩ SMD resistors and a 47nF SMD capacitor, for battery monitoring,
- Power Switch, for convenient on/off without unplugging the battery,
- Piezoelectric Buzzer, for making noises,
- Battery Charger Module, for charging the battery without unplugging,
- Small Rubber Hose, for the feet,
- Any additional sensors or boards you require.
At the time of writing, the required materials could be bought online for about $50 total, but it greatly depends on availability, amount, and quality you require. Also, consider the shipping time, especially for the parts ordered overseas.

### Printed Circuit Board

This is the only custom-made part of the robot. It’s used to connect all the parts electrically and hold them together mechanically. The design files for the board are available in the repository in form of files for the Fritzing program. You can use that program to open the files, optionally make any changes you want, and export them to whatever format you require. Then you can send your designs to a prototype board fabrication service and have them created for you.

If you don’t have an access to a prototype board fabrication service, or don’t want to use it, you can also create the board yourself, or even use a perma-board and make the connections with pieces of wire. This is not covered in this instruction, however.

**Shameless plug.** You can order the board from Dirty PCBs here: order the board. If you do so, I will get a $1. You can also order the board anywhere else, using those gerber files.

### Arduino Pro Mini

The Arduino Pro Mini, or any of the widely available Pro Mini clones, are the cheapest Arduino boards available, even as cheap as $2 a piece. That price is achieved by stripping them of anything that isn’t necessary – including the USB port. You cannot connect this board directly to your computer for programming, as you can other Arduinos, instead you have to use a device called **USB2TTL**. But you only need one such device, and you can re-use it for programming as many Pro Minis, as you want.

You can also use a different small Arduino board, such as Nano or Pro Micro, but that will require adapting the pin numbers, though. They do have an USB port on them, which will stick a little bit out from the back of the robot.

It’s very important that you use a 3.3V version of the board. There are two versions, 5V and 3.3V, which differ in what voltage they expect. If you get the 5V version, it won’t run on the single LiPo battery that we are using here.

The Pro Mini usually comes with three unsoldered pin headers, as you can see here:
You will need to solder at least the short, 6-pin one (it may be a little different version, with pins angled at 90°, but it doesn’t matter much). Remember to make it stick out at the top of the board, on the same side as all the components, and leave the empty side of the board flat. Those are the pins that you will use to program and debug your robot.

If you want to use sockets, you can also solder the two longer pin headers – those come on the bottom of the board, on the empty side. If you don’t want the board to be removable from your robot, you can skip those and use them for the servo sockets instead.

**SG90 Micro Servos**

Those micro servomechanisms are commonly used in radio-controlled toys and models, which makes them easily available and cheap. We will need 12 of them, so the price is really important here. The specific SG90 model was chosen because it’s the cheapest currently available. You can replace it with any other nine gram servomechanism instead.
Each servo comes with a set of plastic horns (single, double and cross-shaped), as well with three screws (one small and two larger). We will need those to build our robot’s legs. This way you don’t need any additional mechanical parts.

Be careful when you are moving the servos by hand, they are not designed for that and it’s very easy to break one of the plastic gears inside them doing that. Such a “stripped” servo is useless for our needs and has to be replaced.

**Pin Headers**

Those are the same as the ones you get with the Pro Mini. In fact, if you are not using sockets, you can use them. But that’s only 24 pins, and you will need a total of 38 pins, so you need to get more. You can buy them easily at any electronics parts shop. Don’t worry about their exact lengths, you will break them into the lengths that you need.
1S LiPo Battery

Your robot will need some kind of power source. We can’t use regular batteries (alkaline or rechargeable), because they are way too heavy for our robot. Remember, that the robot has to be able to carry its own weight on those legs. That’s why we will use a much lighter LiPo battery. You should be able to buy one easily in an electronics parts shop, a hobby remote-control models shop, or you can even use a battery from an old cellphone. Just make sure it’s small and light and has only one cell, that is, has voltage of 3.7V.

Be careful when handling such batteries. Unlike the regular alkaline batteries, they are very sensitive to mishandling. Damaging them mechanically, overcharging or even short-circuiting them can make them explode and release toxic chemicals.
CHAPTER 3

Required Tools

You will need:

- soldering iron and some solder,
- USB2TTL module, with cables,
- cyano-acrylic glue, or other glue suitable for gluing plastic,
- small Phillips screwdriver,
- metal file,
- TV remote.
CHAPTER 4

Assembly Instructions for PCB Version 2

Electronics
Gather all the electronic components that you will need. At the minimum, you will need the printed circuit board, the pin headers and the Pro Mini module. Optionally a 1000µF capacitor, power switch, IR receiver and the battery monitoring circuit consisting of two 47kΩ resistors and a 47nF capacitor, can be added. They are all recommended, so we will assume there are there.

There are two ways of assembling the electronic components for Tote. The first one is compact, like this:

The second one takes up more space, but lets you remove and replace the Arduino module. You will need additional female and male pin headers for that:
Throughout the rest of this guide, we will assume the first, compact, version.

All the electronic components need to be soldered onto the printed circuit board in the right places, as follows:

We will start with the pin headers. First, you have to cut seven pieces, each of them exactly 6 pins long.
You will also need one header that is 2 pin long, for the battery.
Solder six headers for the servo sockets. Make sure they are straight when you are soldering the first pin – it’s almost impossible to straighten them later.
And also solder the battery connector in place. You should have one 6-pin header left. We will need it later.
Now it’s time to solder the battery monitoring circuit. First arrange the elements on the board.
Then carefully solder each one in place.
Finally, solder the remaining header onto the Pro Mini module, so that you can program it.
Now place the Pro Mini on the PCB, and put a drop of solder into every hole.

**Warning:** This is not the best way to get a proper electric connection. After some experimenting, I recommend inserting a piece of wire into the hole before soldering, and then cutting it once it’s soldered in place. Debugging bad connection is really not fun.
To make sure there is a good electric connection, repeat that from the other side.
Now add the power switch. If you have one with 6 pins, just ignore (or cut off) the remaining 3 pins.
Since we are not using a voltage regulator, we need to short the pads that would be used for it. Just solder a small piece of wire in there. You can later remove it if you want to add a voltage regulator.
The 1000µF capacitor has to be soldered with the right polarity. Make sure that the shorter leg (the one marked on the capacitor with a white strip) goes to the hole with a small minus next to it.
Don’t solder the capacitor tightly fitting the board – best leave some wires from the legs (don’t forget to isolate them), so that you can bend it out of the way when needed. Remember, that you will need to screw in the legs in there.
Prepare the IR sensor for soldering. Make sure you check the datasheet for your particular sensor, and bend the legs so that the right ones go into the right holes. In this case, I need to switch two of the legs, so I simply crossed them.
Also leave the legs long, so that you can adjust the position of the sensor later.
That concludes the electronics of our robot.

**Programing**

In order to program the robot, you need to connect it to your computer. To do that, you will need an TTL2USB module, and some Dupont cables (you can also use an FTDI cable, or even another Arduino board). First, connect the cables to your TTL2USB, make sure to note which color goes with which symbol.
Next, connect the other side of the cables to the Pro Mini header. Make sure that the connections are as follows:

- TX0 - RXD
- RX0 - TXD
- VCC - VCC or 5V
- GND - GND
- DTR - DTR

If your TTL2USB doesn’t have a DTR pin, leave it disconnected. You will need to manually press the “reset” button on the Pro Mini when programming it, then.
Now open the `start/start.ino` file in the Arduino IDE, from the “Tools → Board” menu select “Arduino Pro or Pro Mini (3.3V, 8Mhz) w/ ATmega328”, connect the TTL2USB to your computer’s USB port, and click the “upload” button. If your TTL2USB doesn’t have a DTR pin, press the “reset” button on the Pro Mini as soon as the program size appears in the black console at the bottom of the IDE. When you see “Upload finished”, you can disconnect everything and proceed.
Mechanics

Now it’s time to assemble the robot’s legs. You will need 12 small servos, with all the screws and the horns that come with them.
Set all the servos to the zero position (in the middle of their movement range). Then attach the single-arm horn at 90°. You will need six servos with the horn in one direction, and six servos with the opposite. Fasten the horns with the smallest screws (always make sure you un-power the servo before you do that).
Put the legs together.
First, attach the four servos to the body.
Connect them to the outermost servo sockets. Make sure that the yellow wire is inside, and the brown wire outside for all the servos.
Screw the bigger screws into the double servo horn. You will need to use some force, because the holes in the horns are much smaller. Fortunately the plastic is quite soft.
Now attach the single horn to that. Repeat for the other side.
If you want, you can cut off the parts of the screws that are sticking out (or use some smaller screws to begin with). Remove the stickers from the servos. We will need to apply some glue there, and the stickers are in the way.
Attach the horns to two servos. You will need two right legs, and two left ones, that are their mirror images.
Glue the leg to the servo on the body. I recommend an epoxy glue, but as you can see, you can even use a two-sided tape. Make sure the alignment is straight.
Finally, glue a double horn at the end of the leg, as the leg tip. I like to use the slightly longer horn here (the cross-shaped one), as it is lightly longer. Note that two-sided tape doesn’t work very well here, as the area is too small for it to stick properly. Use glue.
You can put a piece of rubber (a small rubber hose works best) at the tip of the leg, to give the robot better traction. Make sure to attach the right legs in the right orientation, as on the image.
Connect all the servos plugs into their respective sockets. The outermost socket for every leg is the hip, then knee, then ankle. Make sure the yellow wire is on the inside, and the brown one on the outside.
Attach the battery on the bottom of the printed circuit board. You can use a rubber band or a double-sided adhesive tape. When connecting the battery, be careful to connect the negative wire to the board’s mass, and the positive one to the switch (there is a small plus printed on the board).
If you want, you can tie the cables together to make the robot look better and to avoid tangling. Make sure you leave enough loose cables for the legs to move freely.
Adjustments

You can edit the `servos.ino` file to adjust the trims for the servos. Edit the `ir.ino` file to change the TV remote protocol, address and button codes, to match yours. You can use the example code from the IRLremote library to read the correct values for your remote. Edit the `leg.ino` file to change the robot dimensions if the parts you used are different from what is described here.
Assembly Instructions for PCB Version 4

Decide on the Version

There are two ways to assemble the board:

You can put everything on top of the board, with the microcontroller board plugged into a set of female headers like this:
Or you can have the microcontroller board permanently attached on top of the robot, and most of the components on the bottom side:
The first approach is recommended when you want to experiment with a lot of different programs and be able to easily swap the microcontrollers, each of them running different code, or when you want to replace the default Pro Mini with a different microcontroller.

The second solution is better when you plan on attaching an additional board to communicate with the Pro Mini and provide higher-level control.

### The Servo Headers

Start by soldering the pin headers for connecting the servomechanisms. You will need six pin male pin headers, with six pins each. You can obtain four of them headers by breaking in half the pin headers that were added to your Pro Mini module, however, you will need at least two more.
Insert the first pin header in its place, and solder the first pin, by touching the soldering iron to the pin and the pad, heating it for a moment, and then touching the pad with the solder.
After soldering the first pin, always rotate the board and check whether the pin header is straight. If it’s not, melt the solder and align the header properly. This will be impossible to fix after you solder the rest of the pins, so remember to always do it.
Repeat that for all six of the headers.

**Battery Header and Optional Jumper Headers**

Prepare two-pin headers for the battery connector and, optionally, jumpers. There are two optional jumpers you can have on the board, marked A and B.

The A jumper lets you disconnect the microcontroller’s power from the rest of the robot. This is useful if you want to power it separately, or don’t want the servos to use power when you are programming the microcontroller from the computer.

The B jumper lets you plug between the power switch with the battery monitoring circuit, and the rest of the robot. Use it if you want to add an additional voltage regulator or boost converter.
If you want to use the jumpers, remember that you have to cut the traces that connect them on the underside of the board. The cutting place is marked with a dashed line:
Note that it may be more convenient to use an angled header for the battery connector.
5.3. Battery Header and Optional Jumper Headers
Optional Microcontroller Socket

If you want the microcontroller board to be pluggable, you will need additional female headers for making the socket. You will need two 12-pin headers, and two 2-pin ones. You can cut longer headers to make them. Start with the long ones:

And then add the two short ones:
Note, that there are two versions of Pro Mini, differing in where the pins A6 and A7 are located. This board works with both of them, just pick the right place for one of the short headers.
Alternative, Solder the Microcontroller Directly

Start by soldering the FTDI headers to the Pro Mini board. Shift the plastic part so that the pins don’t protrude on the back of the module – we want it to be almost flat.
Add additional pins for the battery and serial connections for your extension boards:
Now fill the holes in both the board and the module with solder:
Place the module on top of the board, carefully align it, and melt the solder in each of the holes with your soldering
iron, so that they fuse. Make sure you get proper connections here. It’s recommended to check each connection with a multimeter, and retry if it’s bad.

Optional Battery Monitoring Circuit

Since the battery has higher voltage than the 3.3V our Pro Mini uses, we need to lower it to be able to measure it, using a voltage divider. This circuit consists of two 47kΩ resistors and one 47nF capacitor in the upper left corner of the board:
Place each of the parts in the right spot, and solder one end of it:
When it's in the right position, solder the other end:
Repeat that for each of the parts:
Optional Anti-brownout Capacitor

This capacitor makes sure that the microcontroller has enough current even when the servos momentarily use all the battery can give. It has its place in the upper right corner of the board. Pay attention to the markings on the capacitor and the board, to make sure the negative lead goes into the right hole – electrolytic capacitors are polarized and connecting them wrong may make them explode.
The negative lead is usually marked with a dark strip.
Cut off the excess leads from the other side.

**Power Switch**

The power switch goes on the left edge of the board. Pay attention, as there is a similar set of holes on the other side, but they should be left unpopulated. You can use them later, when extending your robot.
Optional IR Receiver

If you want to control your robot using a TV remote, you will need to add an infra-red receiver. Practically any 38kHz module will do, I have tested TSOP4838, LF0038Q and VS1838B.
The only thing you have to be careful about is to connect the right leads to the right holes: GND to ground, VCC to 3V3, and OUT to pin 2. To do it, you will have to cover the leads with isolation and bend them, as they will most likely be in the wrong order. Check the datasheet of your part for details.
Unlike previous boards, there is no special place to put the sensor. We want it to be connected to pin 2, which is broken out with all other free pins:
Then just solder it in place:
Optional Piezo Buzzer

If you want your robot to be able to make sounds, you can add a piezoelectric speaker on pin 13 – the same as the build-in LED. Any kind will work fine. Cut the leads to a convenient length:
Then solder the speaker to the ground and pin 13:
Glue the speaker under the board.

**Servo Horns**

Insert the single-armed horns that you got with the servos into the large holes in the corners of the board. Depending on the exact model and manufacturer of the servo, the horns may be too big to fit – then you have to use a file to either make the holes a little larger, or the horns a little smaller.
Attach the horns with two screws each:
And then you can attach the leg servo:
Make sure it can rotate freely, without heating the heads of the screws. Use a file if there is a problem.

Note that when attaching the legs to the body, you will need to adjust the angles servo horns – all previous versions of the board had the horns attached diagonally, but this version has them in parallel to the board. That means that you have to remove the screw, remove the horn, rotate the whole leg by 45°, put it back into the horn, and replace the screw. At the neutral position the leg should be sticking out at 45° from the body, shown here:
Legs

Now it’s time to assemble the robot’s legs. You will need 12 small servos, with all the screws and the horns that come with them.
Set all the servos to the zero position (in the middle of their movement range). Then attach the single-arm horn at 90°. You will need six servos with the horn in one direction, and six servos with the opposite. Fasten the horns with the smallest screws (always make sure you un-power the servo before you do that).
Put the legs together.
Screw the bigger screws into the double servo horn. You will need to use some force, because the holes in the horns are much smaller. Fortunately the plastic is quite soft.
Now attach the single horn to that. Repeat for the other side.
If you want, you can cut off the parts of the screws that are sticking out (or use some smaller screws to begin with). Remove the stickers from the servos. We will need to apply some glue there, and the stickers are in the way.
Attach the horns to two servos. You will need two right legs, and two left ones, that are their mirror images.
Glue the leg to the servo on the body. I recommend an epoxy glue, but as you can see, you can even use a two-sided tape. Make sure the alignment is straight.
Finally, glue a double horn at the end of the leg, as the leg tip. I like to use the slightly longer horn here (the cross-shaped one), as it is lightly longer. Note that two-sided tape doesn’t work very well here, as the area is too small for it to stick properly. Use glue.
You can put a piece of rubber (a small rubber hose works best) at the tip of the leg, to give the robot better traction.

Connect all the servos plugs into their respective sockets. The outermost socket for every leg is the hip, then knee, then ankle. Make sure the yellow wire is on the inside, and the brown one on the outside.
If you want, you can tie the cables together to make the robot look better and to avoid tangling. Make sure you leave enough loose cables for the legs to move freely.
Assembly Instructions for PCB Version 5

Those instructions are for version 5 of the PCB. You can tell you have this version, because it’s red and has a “Tote v5” written on it. Start by preparing all the components. At the minimum, you will need these:
Solder the Microcontroller Board Directly

We are going to start with the hardest part – soldering the microcontroller board to the main board. Align the Pro Mini board with the holes carefully, and put a piece of solder wire through the hole:

It’s recommended to start not with the corner hole, like in the photo above, but with the two holes in the center. You can insert some pins into other holes to make sure they remain properly aligned. Now touch the solder wire with your hot soldering iron:
Then cut the wire a couple of millimeters from the board, and repeat for other holes. Now flip the board on the other side, and touch the wires with your iron:
Once you finish all 24 holes, the microcontroller board is soldered.

**The Servo Headers**

You will need six pin male pin headers, with six pins each. You can obtain four of them headers by breaking in half the pin headers that were added to your Pro Mini module, however, you will need at least two more.
The best way to hold the pins in place during soldering is to simply use a breadboard. If you don’t have a breadboard, you can use a sticky tape or simply hold them with your hand (be careful to not burn yourself).
Place the board over the pins, apply some flux to them, and then touch one of the pins with your soldering iron. When it becomes hot, touch the solder wire to the pin, and let it melt and fill the hole around the pin. Repeat for all the pins.
Repeat that for all six of the headers.
Infra-red Sensor

Insert the sensor exactly the way shown on this photo – the order of its pins is important:
Solder it the same way as the pins, then cut off the excess wires:
Power Switch

Bend the legs of the power switch sideways, and place it on the board so that it sticks out a little bit. This time it doesn’t matter which way you place it:
Solder it the same way as the pins.

**Battery Connector**

Depending on the batteries you are going to use, you may either prefer a pin header:
Or just solder the wires directly. Make sure to use the red wire for the plus, and the brown (or black) wire for the minus:
Your finished board should look like this:
Optional Piezo Beeper

If you want your robot to make sounds, you can add a piezo beeper to it:
Since the battery has higher voltage than the 3.3V our Pro Mini uses, we need to lower it to be able to measure it, using a voltage divider. This circuit consists of two 47kΩ resistors and one 47nF capacitor in the upper left corner of the board:

Optional Battery Monitoring Circuit
Place each of the parts in the right spot, and solder one end of it:
When it’s in the right position, solder the other end:
Repeat that for each of the parts:
This is only needed if you are using a LiPo battery. You will also need to uncomment the battery monitoring code in the robot’s program.

### Optional Anti-brownout Capacitor

This capacitor makes sure that the microcontroller has enough current even when the servos momentarily use all the battery can give. It has its place in the upper right corner of the board. Pay attention to the markings on the capacitor and the board, to make sure the negative lead goes into the right hole – electrolytic capacitors are polarized and connecting them wrong may make them explode.
The negative lead is usually marked with a dark strip.
Cut off the excess leads from the other side.

**Servo Horns**

Instead of using the large screws that came with your servos, it’s much better to get some M1.2x4mm screws, and use those. If you still want to use the servo screws, you will need to enlarge the holes in the PCB and in the servo horns. You can do it with a metal file.
Attach the horns with two screws each:
Programming

To program the microcontroller, you will need to connect the USB programmer to the header at the top of the board:
The connections are:

- RX TX
- TX RX
- GND GND
- VCC 3V3
- DTR DTR (if available)

If your programmer doesn’t have a DTR pin, you will need to press the reset button on the board manually, just after your program compiles but before it starts programming. This may require several tries.

I usually don’t solder the header to the programming pins, because you only use them once – you can get good enough connection by just putting them in the holes and pressing with your hand. Once programmed, you can remove them.

To program, open in Arduino IDE the v5/v5.ino file. For assembly and tuning, you will want to edit the file and change the robot_mode variable to 2. This will make the robot simply stand still in its home position, making it easier for you to arrange all the servos properly.

Once you have the legs all fine-tuned, you can re-program it with robot_mode set to 1 for normal walking.
Legs

Now it’s time to assemble the robot’s legs. You will need 12 small servos, with all the screws and the horns that come with them.

You can use the same 1.2mm screws to connect the horns for the femurs:

For the tibias, it’s best to use the original, large servo screw. Cut off on side of the “cross” horn, and use the screw to fix it to the servo’s “ear”.

6.11. Legs
Make sure the servos are all at their center positions, and then assemble the legs by gluing two servos, and putting on the servo horns:
You will need two pairs of legs, one pair normal, and the other the mirror image of the first pair. Attach them to the body at 45° like this:
You will probably need to adjust the horns, so that the robot is in its home position when it starts.
The designs of the main printed circuit board are available in the code repository. You will need Fritzing software to open them and export as whatever format is required for fabricating it.

Arduino Pro Mini is used for the brains of the robot. It is connected directly to the servo signals and to the battery. There is also a 1000µF electrolytic capacitor connected across the battery to avoid brown-outs. An optional voltage divider of two 47kΩ resistors, with a 47nF capacitor for noise reduction is used to monitor the battery voltage and shut down the robot when it drops too low. The battery is any light 1S, 3.7V LiPo cell. Other kinds of batteries can be used, as long as they are light enough and provide enough power. Remote control is done using a VS1838B infra-red sensor, but other sensors can be used.

Several pins of the Arduino are free, including the I²C pins, allowing for connecting more sensors.

The prototype also has an optional LiPo charging/protection module attached, for convenience, although the battery can be charged by any suitable charger.
Tote comes with example program, that makes it walk forward, and lets you control it with a TV remote. If you want to control it from a PC or a different board, then you will have to write your own software.

All the required code is included in the repository. You will need an Arduino IDE and a TTL2USB programmer for the Pro Mini to compile and upload the software into the robot. Connect the programmer, open `start.ino` in the Arduino IDE, and click “Upload”. That’s all.

For the infra-red communication, I’m using the excellent IRLRemote library, written by NicoHood. You may want to adjust the `ir.ino` file to modify the button commands and the protocol, so that they match your TV remote.

## Features

### Creep Gait

Right after switching on, the robot will start walking forward using the creep gait. This is so that you can see that it works properly even when you don’t have a TV remote at hand.

### Remote Control

You can use a TV remote to control the robot. Direction keys will make it speed up/slow down, go backwards or turn. You can also raise and lower its body, spread the legs more, make it go sideways, etc.

### Power Off

The power button on the TV remote will make it fold and shut down – the servos will be powered down and the board will enter a sleep mode. You should switch it off with the power switch manually anyways, because it will still consume some power in that state.
Battery Monitoring

If the battery voltage drops too low (and you have soldered the optional two resistors and a capacitor required for monitoring it), it will also fold and switch off when the battery voltage runs too low. Note, that it will probably not have enough power to finish folding in this situation, so it may end up in an awkward position, or even in a restart loop. You should switch it off and charge the battery as soon as possible.

Trot Gait

Pressing the direction buttons multiple times makes it walk faster by making longer steps, until it can’t go faster anymore, and then it will switch to a trot gait, which is about two times faster, but unstable.

Automatic Leg Order

The order in which the legs make steps is important for stability. The robot will automatically select the optimal leg order depending on the direction in which it is moving.

Beeper

If you connect a piezo buzzer to pin 13 and ground, you can make this robot beep with every step and when it receives a command. This is muted by default and has to be enabled with a button on the remote, because it’s so annoying.

Modes

You switch the modes of operation of the robot with a button on the remote. Initially there are only two modes: walking and doing nothing. You can easily add your own modes for additional functions that you program.
Since this robot is so simple, it is very easy to modify or extend it. You can add sensors and peripherals, or control it remotely from another computer or directly with another microcontroller or microcomputer board.

**GPIO Pins**

Most of the Arduino’s pins are used internally, but there are several of them left for your use:

- 7 pins can be used as digital,
  - 1 of them has a LED on it,
  - 2 of them can be used as serial interface,
  - 2 of them can trigger interrupts,
    * 1 of them has PWM,
  - 2 other can be also used as analog pins,
    * they can also be used as I²C interface,
- 2 analog-only pins.

**Sensors**

You can connect anything that runs on 3.3V and can be connected either to the free GPIO pins, or to additional board or multiplexer that you add. Examples include:

- ultrasonic distance sensor, such as HC-SR04,
- switches at the ends of the feet,
- accelerometer, gyroscope and/or compass,
- encoders for leg positions,
• light sensors, for following light,

**Peripherals**

You can also use the GPIOs to attach additional hardware that would do something. Unfortunately, you cannot easily add more servos, as the Arduino Servo library being used only supports 12 of them at once. But instead you can add:

• piezoelectric buzzer, for beeping,
• addressable LEDs, for light effects,
• infrared diode, for “shooting” at other robots in a laser-tag game,
• a nerf gun,
• a LiPo battery charger.

**Remote Control**

You can use the serial interface to connect a transmitter/receiver module and make the robot communicate with another device. For example, you can use:

• WiFi module, such as ESP8266,
• Bluetooth module, such as HC-05,
• Radio module, such as NRF24,
• BLE module,
• and so on.

**Additional Boards**

You can also add another microcontroller or microcomputer board to the robot, and make it communicate with it either through the serial or I²C pins. Some examples include:

• Arduino Pro Mini, with more pins,
• VoCore, a computer running OpenWRT Linux,
• PyBoard, a microcontroller running MicroPython,
• Teensy 3.1, an ARM-based microcontroller board,
• OpenMV, a microcomputer with a camera, programmable with Python,
• ESP8266 with NodeMCU firmware, programmable in Lua.

The possibilities are infinite, as long as the board in question has a serial or I²C interface, can be powered from a single LiPo battery, and doesn’t weight too much for the robot to carry it.

If your board or module can be powered with 3.3V, simply use the FTDI header of your Pro Mini for connecting it – it has broken out the power and serial interface. You will have to write the code for communicating with the board yourself.

If the board requires 5V or considerable amounts of current, you don’t want to power it from Pro Mini’s power regulator. Instead, add pins to the RAW and GND pads on it, and to the RX and TX pads:
Then use a double-row female header to connect your breakout board to the Pro Mini:
And connect your board on top of that:
Of course, you will need to use a power source that can power your board.
Design Goals and Constraints

The main goal is to have a walking robot that is interesting to program and at the same time easy to build. From that, we get a number of constraints:

- The robot has to fit on the desk, because not everyone has a garage.
- It has to have inverse kinematics, otherwise we are limited to replaying pre-recorded movements.
- It should avoid parts that are custom-made using tools and materials that are not widely available. In particular, no laser-cutting or 3D-printing.
- The parts should be common, cheap and easy to get, also in bulk.
- It should be possible to easily extend or modify the robot.
- In the basic version it should have enough functionality to be fun to play with.

Hacks and Workarounds

Because of the above constraints and the desire to make it all as cheap as possible, I’ve made some design decisions that are not exactly good engineering. I figured out that it’s better to have a crappy but working robot, than a beautifully designed plan. So here are some corners that I’ve cut:

Mechanical Design

The way this robot is assembled together only works because it’s so lightweight. Would the robot be heavier, there would be significant problems:

- Servomotors are used as structural elements.
- The printed circuit board is also used as a structural element.
• The whole robot weight rests sideways on the servo shafts.
• The servo horns used for legs are elastic.
• The screws that come with the servos are too big for what they used for.
• The servo cables are all over the place.
• The power switch is in an awkward place.
• There is no good way to attach the battery.
• The legs need some rubber at their ends, or they will slip.

Most of those problems can be solved by using better materials and adding some structural support elements, but that affects the price and complexity.

**Electrical Design**

The desire to cut costs and minimise the number of parts also made me make some not-entirely-kosher decisions on the electrical side of things:

• The servomechanisms are under-powered. The specification asks for 4.6V, and they get from 3.6V to 4.2V depending on the charge level of the LiPo battery.
• The servomechanisms are powered directly from the battery, without any kind of voltage regulator.
• The undercharge battery protection is software-based, which means it can easily fail.
• There is no battery charger build in. You have to disconnect the battery to charge it.
• The Pro Mini is soldered directly onto the printed circuit board.

Those problems can be easily solved by adding the missing parts to the design. There is even some room left for them, in the form of additional holes and pins.
Kinematics is the process of determining how a robot will move given certain inputs. Inverse kinematics is for determining what inputs we need to give it to get into a certain position. In particular, leg inverse kinematics for walking robots is the process of calculating the angles for all the joints required to get the tip of the leg in a certain position relative to the body. There is also body inverse kinematics, which tells us in what positions the legs have to be for the body to be in a certain position (tilt, yaw, rotation, shift).

This are notes for determining the inverse kinematics for a four-legged robot with three degrees of freedom per leg. Let’s start with an image of the leg:
Basic Trigonometry

For a start, need to determine the geometry of the leg and describe it using mathematical formulas. For that, we will start with a few tricks from the elementary school.

To calculate the diagonal of a rectangle, we can use the Pitagoras theorem, \( c^2 = a^2 + b^2 \), so \( c = \sqrt{a^2 + b^2} \).

To calculate one of the angles of a triangle for which we have the lengths of all sides, we can use the Law of Cosines: \( c^2 = a^2 + b^2 - 2ab \cos \alpha \), which gives us \( \alpha = \arccos \frac{a^2 + b^2 - c^2}{2ab} \).

Finally, to calculate the angle of the diagonal in a rectangle, we have \( \beta = \arctan \frac{y}{x} \), or better, \( \text{beta} = \text{atan2} (y, x) \), as most languages have a convenience function for this that also handles the sign properly.

Solving a Leg

We are given the coordinates \((x, y, z)\) and we want to calculate the angles \((\alpha, \beta, \gamma)\) of the hip, knee and ankle joints. We also have the lengths of coxa \(L_c\), femur \(L_f\) and tibia \(L_t\). The hip is easy:

\[
\alpha = \text{atan2}(y, x)
\]

All the rest is happening in one vertical surface along the hip angle, so we can convert our \((x, y, z)\) into \((f, z)\), where \(f = \sqrt{x^2 + y^2} - L_c\) is the part of the diagonal starting at the knee joint (see picture).

The knee angle is going to consist of two parts. The first is easy, \( \beta_1 = \text{atan2}(z, f) \). For the second, we need to know the length of the diagonal \(d = \sqrt{f^2 + z^2} \). Once we have \(d\), we can solve the triangle formed by tibia and femur, and we get \( \beta_2 = \arccos \frac{L_t^2 + d^2 - L_f^2}{2L_td} \). Finally:

\[
\beta = \beta_1 + \beta_2
\]
Solving the same triangle, but for a different angle, will give us our ankle angle:

\[
\gamma = \arccos \frac{L_f^2 + L_t^2 - d^2}{2L_f L_t}
\]

**Trim and Reverse**

We have the angles, but they are unlikely to have to align with where our physical servomotors have their zero angles, and the direction they turn. So we have to add some offsets to them, and possibly reverse them. The easiest way to go about that is to set all your servos to 0 and check where the end of the leg should be then. Then run those coordinates through our formulas, and you will get the numbers that you have to subtract from our calculated angles (so that at the initial position they are all zero). Then move them a little from the initial position and see if the directions are right.
The way in which an animal or a robot walk is called a gait – it tells us how and in what order the legs are moved, how the body is balanced and how it moves forward.

**Statically and Dynamically Stable Gaits**

When the robot walks, it has to keep its balance. There are two general strategies for doing that, and according to them, we divide the gaits into statically stable and dynamically stable.

For statically stable gaits it doesn’t matter how fast they are performed, or whether the robot is stopped in a middle of a step – it is stable at any moment, at all times. Animals and people use those gaits when they want to go slowly, or when they want to be able to stop at any time. An example of such a gait is the “creep” gait, used by cats stalking their prey.

Dynamically stable gaits are much harder, as they have to be performed at a particular speed and cannot be interrupted at an arbitrary point. They are sometimes called “controlled falling”, as they exploit the fact that it takes some time for the robot to fall when it’s unstable, and that time can be used to move the legs in such a way as to prevent the fall. Most animal gaits are dynamically stable, as they tend to be faster and more energy-efficient. An example of a simple dynamically stable gait is the “trot” gait.
I order for a relatively light robot, like ours, to be statically stable, it has to keep its center of mass somewhere between its legs. More precisely, if you connect all the feet on the ground with straight lines, so that they form a convex polygon, the center of mass has to be located directly over it. We call that polygon the “area of support” of the robot at the given moment. The area of support will change depending on which legs are placed on the ground and where, so the robot will sometimes need to shift its center of mass around to remain stable while changing its stance.

The image above shows the area of support of our robot in the starting position.

In a heavier robot, you also have to take into account the inertia, so that it’s not sufficient to just track the center of mass – you have to track the so-called “zero-moment point”, often abbreviated ZMP, which is basically the point on the ground directly below the center of mass, shifted to account for the inertia. We will not get into details of that here, as our robot is light enough that just leaving some small margin around the area of support is enough.

Raising a Leg

When the robot raises one of the legs, its area of support changes dramatically:
You can see, that our center of mass is directly on an edge of our area of support. That’s bad, because the smallest force can now tip our robot and make it fall. We can avoid that by moving all the legs toward the one leg that we want to raise before raising it, so that the center of mass shifts:
There are also other methods of balancing, not covered here. For example, many animals have a flexible spine, that they can bend sideways to shift their center of mass. Other animals have tails that they use to balance. Possibilities are endless.

**Moving Forward**

Our goal is to propel our robot horizontally across the floor. The simplest solution to do that, at least initially, is to just move all the legs that are touching the ground backwards. If we do it slow enough, so that they don’t slip, they will stay where they were, but move the body of the robot forward. This is an excellent way of moving around, and many industrial robots do it this way, bolted to the floor in one place. There is however a small problem: the range is quite limited. Sooner or later you will either move your center of mass outside the support area, or your legs move to their maximum range. What then?

Then you raise the legs from the ground, move them forward, and set them down in the new place. But you have to be smart about it, because if you do it with all the legs at once, you will fall down. But you can raise a single leg and still stand on the remaining three. You can also raise and move that leg *while* moving the remaining legs backwards, which saves you some time and makes it both faster and more smooth.

Now, as you know, if you are raising a leg, you also have to make sure that the center of mass stays inside the support area. To do that, you will have to shift the robot’s body away from the leg that you are rising. Since you are shifting your body forwards, it is also important in which order you raise the legs – some orders will be more stable than others.

So, to sum, our plan of action is:
• move all the legs that are on the ground backwards,
• for each leg in a set order:
  – shift the body away from that leg
  – raise the leg
  – move the leg forward
  – set the leg on the ground.

The Creep Gait

What order is the best for making steps, if you raise one leg at a time? There are four possible orders, assuming you always start with the same leg, and ignore the orders that are just reflections.

Both calculations and experimental data show that the last one, a figure eight orthogonal to the direction of walking, provides the optimal stability at all moments. Note that this order is best only when you are moving forward. If you are moving sideways, backwards or rotating, then other leg orders are optimal. That’s why it probably makes sense to adapt the order to the situation.
The Trot Gait

The creep gait is stable, but quite slow – raising only one leg at a time, you can’t move the remaining legs backwards too fast, or you move out of the range of your legs. In addition, shifting the robot’s body before each step also takes some time. Is there a faster way?

Turns out that you can move two opposite legs at the same time, if only you do it fast enough, so that the robot will not have time to tip on the two remaining legs. This is not a statically stable gait – you can only stop safely between steps, when all the legs are on the ground, but it’s more than twice as fast as the creep gait: first, you move two legs at a time, and second, you don’t have to shift your body for balance.
This tutorial should take you through all the steps you need to design a walking robot like the Tote. It’s for people who are not satisfied with just blindly following a ready blueprint, but would rather want to be able to design their own robots. This takes considerably more time, than just assembling a robot, but it can be done one small step at a time.

**Soldering the Pin Headers**

In order to be able to connect your Pro Mini to your computer, you first need to solder the pin header for the serial communication. Your board should have come with three pin headers: one 6 pins long, and two 12 pins long. You will need the shorter, 6 pin long one. The pins in it may be straight, like in the illustrations, or they may be angled – it doesn’t matter for our purposes.

Insert the pin header into the holes at one of the short ends of the board, as the illustration below shows:
Now flip the board to the other side, place it on a table, and carefully solder each of the pins to their holes. To do that, first touch the hole with the end of your solder wire, and then briefly touch it with a hot soldering iron, as shown below:
The solder should melt and flow into the hole, filling it and securing the pin in place. Make sure you don’t connect the neighbouring pins with each other while doing this. Each of them should have a separate electric connection only to their own hole.

**Connecting the Pro Mini to the Computer**

Unlike most other Arduino boards, the Pro Mini cannot be directly connected to your computer’s USB port. Instead, you will need to use a small device called USB2TTL adapter, which basically contains all the electronics required to support a USB connection that the Pro Mini is missing. This makes the Pro Mini much cheaper than the other boards, and you can re-use the adapter for other projects. You will also need a cable to connect the USB2TTL with your board. It’s simply 5 wires, with small sockets on both sides, which slide onto the pins that you just soldered, and onto the pins of the adapter. You have to pay very close attention to which pins you are connecting, though. Look at the symbols on both boards, and connect the pins like this:

- The DTR pin on the Pro Mini connects to the DTR pin on the adapter.
- The TXD (or simply TX) pin connects to the RXD (or RX) pin.
- The RXD (or RX) pin connects to the TXD (or TX) pin.
- The VCC pin connects to the 5V (or VCC) pin.
- The GND pin connects to the GDN pin.
If your adapter lacks the DTR pin, don’t worry, you can skip it, but then you will have to manually restart your board each time you are uploading code into it.

Once you made all the connections, insert the USB2TTL into your computer’s USB port. If everything went right, and your Pro Mini is brand new, you should see one LED on it light up (it’s the power indicator), and another LED blink about once a second. This is the default “demo” program that is uploaded to all new Arduino boards, and it shows that your board is working.

**Uploading First Arduino Program**

In order to actually write and upload programs onto your board, you will need the Arduino IDE software. You can download it from the Arduino website for your platform of choice. Refer to the information on that website for help with installing and configuring it.

Once you have that working, you can select the “blink” example from the examples menu, select the right board type (Pro Mini 3.3V), connect your board and hit the “upload” button. With a little luck you will have your own “blink” program uploaded. To verify that it works, you can change the delay between blinks, and upload it again.
Blinking a LED from the Computer

Attaching a Buzzer and Making Sounds

Attaching an RC Servo

Controlling the Servo and Buzzer from the Computer

Attaching an Infra-Red Sensor

Blinking a LED, Making Sounds and Moving Servos with a TV Remote

Making a Pan-and-tilt Gimbal

Attaching a Battery

Assembling the Printed Circuit Board

Monitoring Battery Voltage

Quadruped with Two Degrees of Freedom per Leg

Leg Inverse Kinematics

Unbalanced Crawling

Crawling Sideways

Rotating in Place

Turning While Crawling

Changing Body Height

Shifting the Body

Balanced Crawling

Trotting

Attaching a Proximity Sensor

Following an Object
Feedback and Help

Feedback

If you build this robot, or, especially, if you improve on the design or extend it, please let me know! I would love to hear about it! Perhaps your improvements can be even incorporated in this guide (with your agreement).

Help and Support

This guide is provided as is, and I can’t be responsible for what you actually build using it, and whether it works as you expected or not. No guarantees. But legalese aside, I would really like to help, so please don’t hesitate to contact me if you get stuck at some point or need advice. I am an amateur myself, but I will try to help as much as my time and knowledge allow.

Contact

There are several ways you can contact me about this guide. First, if you find an error, a typo, an omission or generally know of a way of improving this guide, please either send a pull request to the repository, or report an issue in it. This is the fastest way of having it fixed.

If you want to talk to me, I can be often found on the #robotics channel on the Freenode IRC network, and also on the letsmakerobots forums. In both of them I use the nick deshipu, but there will also be a lot of other very friendly people that will be able to help you.

Finally, if you are interested in my other projects, or would like to contact me privately, visit my home page.