Tightrope Unicycle Bot
Created by John Park

Last updated on 2020-12-17 10:53:59 AM EST
Overview

Adabot has always wanted to join the circus -- and you can help Adabot fulfill this dream by building a tightrope riding, balanced unicycle bot!

This project uses the Circuit Playground Express for brains, and the Crickit platform to provide the power to the gearbox DC motor.

You'll make a cardboard chassis, frame, and armature, then dress it up with Adabot costuming.
Parts
These are the parts you'll use to build your bot.

1 x Adafruit CRICKIT
for Circuit Playground Express
Add to Cart

1 x Circuit Playground Express
Round, Awesome Microcontroller Board
Add to Cart

1 x TT Motor Bi-Metal Gearbox
1:90 Gear Ratio
Add to Cart

or, you can use the yellow TT motor, it can’t go quite as slow, but works well, too!

1 x TT Motor DC Gearbox
Plastic gear -- 1:48
Add to Cart

1 x TT Motor Pulley
36mm Diameter
Out of Stock

1 x Waterproof 3xAA Battery Holder
with On/Off Switch
Out of Stock

1 x Male DC Power adapter
2.1mm plug to screw terminal block
1 x Alkaline AA batteries
3 pack

1 x Mini Metal Speaker w/ Wires
8 ohm 0.5W

1 x White Nylon Screw and Stand-off Set
M2.5 diameter

Out of Stock

Materials
In addition to the parts above you'll need

- corrugated cardboard
- hobby knife or scissors
- small Philips screwdriver
- string or cord
- brass paper fastener
- white glue or hot melt glue and a glue gun
- two M2.5 x 12mm screws and nuts
- an M3 x 8mm screw
- a toothpick

For the Adabot costuming overlay, you'll need either a color printer and some thick printer paper, or cardstock in light blue, royal blue, white, and black to make the cutout layered version designed by the Ruiz brothers.
Build The Robot

Your robot will be made from two sections, each connected to the TT motor at the center of mass. The motor will drive a pulley wheel that rests on the tightrope.

The chassis section will hold the battery pack and act as the mounting point for the Crickit and CPX.

The upper frame will support the robot armature and costuming parts.

Let's get building!

Make the Chassis
Cut a piece of corrugated cardboard from a shipping box, about 16"W x 12"H

Use the template below (or a Cricut or laser cutter) to cut out the four parts.

Template
Download the .svg or .pdf file depending on your preferred tools. You can print either one and then use it as a template on your cardboard. You can cut out each shape, then trace around it with a pence while you hold it to the cardboard.

Then, cut out the shapes. Be careful not to cut along the purple dashed lines -- those are just to indicate
where you'll fold the cardboard.

Be sure to print the template at 1:1 size, don't let your printer software scale it! You can print in tile mode to create properly scaled printouts if the image is too large for a single page on your printer.

Chassis Assembly
• Place four nylon standoffs into the cardboard frame as shown and secure them with nuts (you can do all eight if you like, but it's overkill)
• Screw the Crickit onto the standoffs
Battery Box Holder

Put three fresh alkaline AA batteries in your battery box before proceeding.
Fold the chassis at the two fold lines to form the battery holder.
Fold the two arms at their fold lines.
Place a brass fastener into the holes in the arms.
Set the battery box in place.
Secure the arms using the brass fastener.
Motor Prep (for motors without wires already)
Next, we'll prepare the motor by soldering two wires to the tabs, and screwing the pulley wheel onto the shaft.

If you're using the yellow, plastic geared TT motor, you can skip this next step, as it comes with wires pre-soldered to the motor's tabs.
- Strip the insulation from the ends of two short wires -- around 5" in length
- Solder a wire to each tab of the motor
- Polarity doesn't matter, since we can insert either wire into the motor terminals on the Crickit board, as well as choose forward and backward direction in software later

If you're using the yellow, plastic geared TT motor, use the provided screw that come with the pulley wheel. No need to widen the pulley wheel center hole, either!
• Widen the center hole of the pulley wheel a little bit with a small drill or the tip of a screwdriver
• Attach the pulley wheel to the shaft of the motor with an M3 screw
Upper Frame
The upper frame will connect to the motor and chassis in order to provide a structure from which to hang the robot.

- Use a straight edge to crease the upper frame piece along the bend marks
- Fold the two bends at 90° each
- To help the cardboard hold its shape, you can add
some white glue or hot melt glue to the insides of the creases, then hold it in place while the glue sets

Assemble the Parts
Now, its time to assemble the chassis, motor, and upper frame to each other. You can use two long M2.5 screws and nuts, or if you don't have those on hand, use zip ties, or even string!
Screw the battery box’s red and black wires into the DC plug adapter’s ‘+’ and ‘−’ terminals, respectively.

Connect the Chassis, Motor, and Frame
Use the long M2.5 screws to attach the chassis to the motor as shown.

Attach the frame to the screws on the back-side of the motor.

Use the two nuts to fasten.

- If you chose to use a yellow TT motor, you'll also need to poke out a small hole for the rear axle to poke through the upper frame.

Wire the Motor

Next, run the motor wires to the Motor 1 terminals on the Crickit board and screw them in place. You can insert either wire in either of the two terminals on the Motor 1 block -- you do not need to use the terminal marked GND.
Center of Mass

Try holding the robot with one finger under the pulley wheel -- you'll see that it balances pretty well. This is because most of the weight of the Crickit, CPX, and batteries is directly underneath the pulley, with a bit of it protruding beyond it in order to compensate for the weight of the motor itself.

This means that pulley wheel is at the robot's center of mass (also called "center of gravity"). This is what will allow it to balance on the tightrope without tipping over.

Speaker

To give your robot some atmospheric sound capabilities (big top music, perhaps!) screw the two speaker wires into the Crickit's speaker terminals, then secure the speaker to the chassis with tape, glue, or poster...
putty.

Leg Armature
Next, we'll build the leg. The leg will rotate around two pivot points, the hip and knee, and connect to the pulley wheel to provide rotation of the system.
Knee Joint
overlap the lower and upper leg segments at the 'knee' joint
- put a nylon screw through the knee holes
- secure with a nut -- but not too tightly, this joint needs to rotate freely!

Hip Joint
• Place the upper leg pivot hole over the torso's hip pivot hole
• Gently secure it in place with a nylon screw and nut, again, making sure it can rotate freely

Ankle Joint
If you have a nylon screw that's long enough to use as an ankle pivot, go ahead and use it. If not, have no fear, we can make the ankle pivot from a toothpick!
• Cut off 1/3 of a toothpick with some scissors
• Insert the toothpick through the ankle pivot hole and through one of the outer holes on the pulley wheel
• Allowing the toothpick to protrude a bit from either end of the mechanism, trim it to size
• If you worry the toothpick pivot may fall out, you can flatten each end a bit like a makeshift rivet
You've completed the structural, electrical, and mechanical build for your robot! Next, we'll program it, and later we'll dress it up a bit!

Try hand turning the pulley wheel to see the leg go through its motion range.
Program the Robot
Coding in CircuitPython for Crickit and Circuit Playground Express
Crickit and Circuit Playground Express (CPX) are a great combo for robotics projects. And when paired with CircuitPython, it's very straightforward to write your own code and iterate quickly during testing.

Setup the CPX and Crickit
The Circuit Playground Express (CPX) paired with the Crickit is a powerful, yet simple to use combination for building animatronics! To get started, you'll want to set up the CPX for use with CircuitPython by following this guide (https://adafru.it/AFI). When you're ready, and can upload code to the board return here.

To use the Crickit with the CPX, follow the steps listed here (https://adafru.it/Bfh) to install the special build of CircuitPython (name will begin with 'adafruit-circuitpython-circuitplaygroundexpress-crickit...'), as well as the latest library bundle.

Audio File
We'll have the bot play some circus music when it starts up. Download the file here and copy it to the root level of your CPX CIRCUITPY drive that shows up when it's plugged into your computer.

https://adafru.it/BkE
The CC licensed circus theme music came from Freesound (https://adafru.it/BkF).

CircuitPython Code
Adafruit suggests using the Mu editor to edit your code and have an interactive REPL in CircuitPython. You can learn about Mu and installation in this tutorial (https://adafru.it/ANO).

The full code is included below -- you can copy and paste that into Mu and save it to your CPX to get it all working. After the full code is an explanation of how the various code snippets work.
import time
import board
import neopixel
import audioio
import adafruit_crickit

print("Adabot Tightrope Unicyclist!")
RED = (16, 0, 0)
GREEN = (0, 16, 0)
BLACK = (0, 0, 0)

pixels = neopixel.NeoPixel(board.NEOPIXEL, 10, brightness = 0.2)
pixels.fill(BLACK)

# Create a motor on Crickit Motor 1 port
motor = adafruit_crickit.crickit.dc_motor_1

# User variables
run_time = 6
speed = 0.65

# Music

cpx_audio = audioio.AudioOut(board.A0)  # speaker out on Crickit
def play_file(wavfile):
    with open(wavfile, "rb") as f:
        wav = audioio.WaveFile(f)
        cpx_audio.play(wav)
        while cpx_audio.playing:
            pass

wav_file_name = "circus.wav"
play_file(wav_file_name)

while True:
    # set NeoPixels green in direction of movement
    for i in range(5):
        pixels[i] = GREEN
    for i in range(5):
        pixels[i+5] = BLACK

    motor.throttle = speed  # full speed forward
    time.sleep(run_time)  # motor will run for this amount of time

    # set NeoPixels red when stopped
    pixels.fill(RED)
    motor.throttle = 0  # stop the motor

    # set NeoPixels green in direction of movement
    for i in range(5):
        pixels[i] = BLACK
    for i in range(5):
        pixels[i+5] = GREEN

    motor.throttle = -1 * speed  # full speed backward
    time.sleep(run_time)  # motor will run for this amount of time

    pixels.fill(RED)
    motor.throttle = 0  # stopped

Here are the different sections of the code, and how they work.
Library Imports
First, we'll import a few libraries. We'll import `time` so we can use the sleep command.

We also import the `board` library, which gives us convenient definitions for objects on the CPX that we can call.

Then, we'll import `neopixel`, `audioio` and `audiocore`. That way we can blink lights (neopixels) and make some music!

Finally, the good stuff - from the `adafruit_crickit` library we import `crickit`, this is the library that lets us talk to the Crickit robotics board and make stuff move!

These code snippets aren't meant to be run on their own, they are here for explanation purposes.

```python
import time
import board
import neopixel
import audioio
import audiocore
from adafruit_crickit import crickit
```

NeoPixel Prep
We'll define variables to represent the color values we'll use for the NeoPixels to indicate forward, backward, and stopped states.

Then, we'll set up the NeoPixel object and set them to black (off).

```python
RED = (16, 0, 0)
GREEN = (0, 16, 0)
BLACK = (0, 0, 0)

pixels = neopixel.NeoPixel(board.NEOPIXEL, 10, brightness = 0.2)
pixels.fill(BLACK)
```

Motor Prep
The next thing to do is to instantiate the motor output, we'll be using the Motor 1 port (The two pins are labeled on the Crickit as Motor and then the "1" pair) To make things easier to type we'll call the motor, simply `motor`!

```python
# Create a motor on Crickit Motor 1 port
motor = crickit.dc_motor_1
```

User Variables
Depending on your tightrope's length, you may want to unicycle for greater or lesser amounts of time in each direction. You may also want to direct Adabot to go faster or slower.

These two variables -- `run_time` and `speed` -- are used to define the run time in each direction in seconds, and the amount of power to send to the motor from '0' to '1'. (Technically, this goes down to '-1' for reverse, but we'll handle that with a bit of math later.)
Music, Maestro
This next section is used to create an audio output object named `cpx_audio`, and then to define a process called `play_file()` we can call upon to play back audio files in the .wav format.

```python
# set NeoPixels green in direction of movement
for i in range(5):
    pixels[i] = GREEN
for i in range(5):
    pixels[i+5] = BLACK
```

We saved a file to the CPX named 'circus.wav' so we'll cast a variable for that, and then run the `play_file()` command to have it play once during startup of the robot.

```python
wav_file_name = "circus.wav"
play_file(wav_file_name)
```

Main Loop
With the setup complete, now we get into the bulk of the program, which runs over and over. This is everything that follows the `while True:` line.

The main loop does just a few things over and over:

- Light the five NeoPixels on the left of the board green
- Run the motor forward for a set amount of time
- Light all NeoPixels red
- Stop the motor
- Light the five NeoPixels on the right of the board green
- Run the motor backward for a set amount of time
- Light all NeoPixels red
- Stop the motor
- Repeat

This is the code that does those steps:

NeoPixels Forward
This runs through the first five NeoPixels in the `for` loop and sets them to green, then runs through the second five and sets them black, or off.

```python
# set NeoPixels green in direction of movement
for i in range(5):
    pixels[i] = GREEN
for i in range(5):
    pixels[i+5] = BLACK
```
Motor Forward
This section uses the `motor.throttle` command and sets it to the `speed` we previously defined, for the `run_time` in seconds.

```
motor.throttle = speed  # full speed forward
time.sleep(run_time)  # motor will run for this amount of time
```

Stop
Then, we set all NeoPixels red, and stop the motor.

```
# set NeoPixels red when stopped
pixels.fill(RED)
motor.throttle = 0  # stop the motor
```

Then, we essentially repeat these steps for the reverse direction. Note, the use of `motor.throttle = -1 * speed` as a way to reverse motor direction.

```
# set NeoPixels red when stopped
pixels.fill(RED)
motor.throttle = 0  # stop the motor

# set NeoPixels green in direction of movement
for i in range(5):
    pixels[i] = BLACK
for i in range(5):
    pixels[i+5] = GREEN

motor.throttle = -1 * speed  # full speed backward
time.sleep(run_time)  # motor will run for this amount of time

pixels.fill(RED)
motor.throttle = 0  # stopped
```

Once you start to run your unicyclist on the tightrope you can come back and revisit the code to change the behavior.

- What if you want to pause between directions and play the song again?
- How about varying the speed to ease into and out of the top speed?
- Since the same things happen in each direction, can you optimize the code to reduce the repetitive code blocks?
Add Character

Just like an animatronic robot at a theme park, your unicycle robot has a functional skeleton that can be covered over with an outer skin or costuming.

A simple way to do this is with cutout Adabot parts! We've provided two templates you can use to do this in a couple of different ways.

With the first, you can simply print to a color printer and then cut out the parts. The second file is for layering pieces of colored cardstock that are either cut out with a hobby cutting machine, such as a Cricut paper cutter (https://adafru.it/BkR), or with a laser cutter.

Printer File
The .pdf file below is for printing out to a color printer and then cutting out the parts.
Cutter File
This file is the one to use if you have a machine cutter or laser cutter and you want to layer cardstock.
Glue and layer the light blue leg and arm parts on top of their dark blue counterparts.

For the head, the layer order is: white square (this provides white for the eyes and teeth), dark blue, light blue, and finally the black eyes.

You can attach the head to the torso by putting some glue on the neck protrusion and affixing it to the back side of the torso.

**Dressing the Bot**
Now, you can use tape or glue to attach the costuming to the underlying cardboard parts.
Upper Leg
• Place a glue dot, tape, white glue, or hot melt glue to attach the middle of the upper leg costume to the middle of the cardboard upper leg
• Make sure the leg can still pivot freely at the hip and knee

Lower Leg
Attach the lower leg costuming in the same way, making sure not to impede the pivots.
Torso and Head
Next, affix the torso and head to the cardboard body armature.

Arms
You can decide how to affix the arms to provide that perfect look for your bot! Here they're taped in place, but what if you wanted to be able to change the arm pose? You could add some more holes and nylon screw pivots! This is one of the great things about working in cardboard and paper -- it's never too late to make modifications!
Your tightrope unicycle Adabot is ready to perform!
Get on the Tightrope

Your robot is ready to take to the tightrope! Loop a long piece of cord or string around two sturdy objects -- in a classroom you can try using two legs of a long desk. Especially dramatic if you flip one table upside down on top of another!

Tie off the cord securely, keeping things from getting too loose. This can be tricky. A fisherman’s knot (https://adafruit.it/Bkl) is a good, simple choice for this, as it's made with just two overhand knots.

Now, feed the bot up from the bottom between the two cords, and then hook the pulley wheel over the cord closest to you.

Note: shown here with the alternate yellow TT motor!
Flip the Crickit's on/off switch to ON! You'll hear the circus theme music play and then Adabot will start to unicycle back and forth on the tightrope!