# LAPENNA Program
# WEBINAR 7 – RC Vehicle

**Kwai Wong, Stan Tomov**
**Rocco Febbo, Julian Halloy**

**University of Tennessee, Knoxville**

**October 16, 2020**

# References

- https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit
- https://elinux.org/Jetson_Zoo
- https://www.arduino.cc/en/Main/Software
- https://www.balena.io/etcher
- https://bitbucket.org/CFDL/opendnnwheel/src/master/
- https://imageai.readthedocs.io/en/latest/

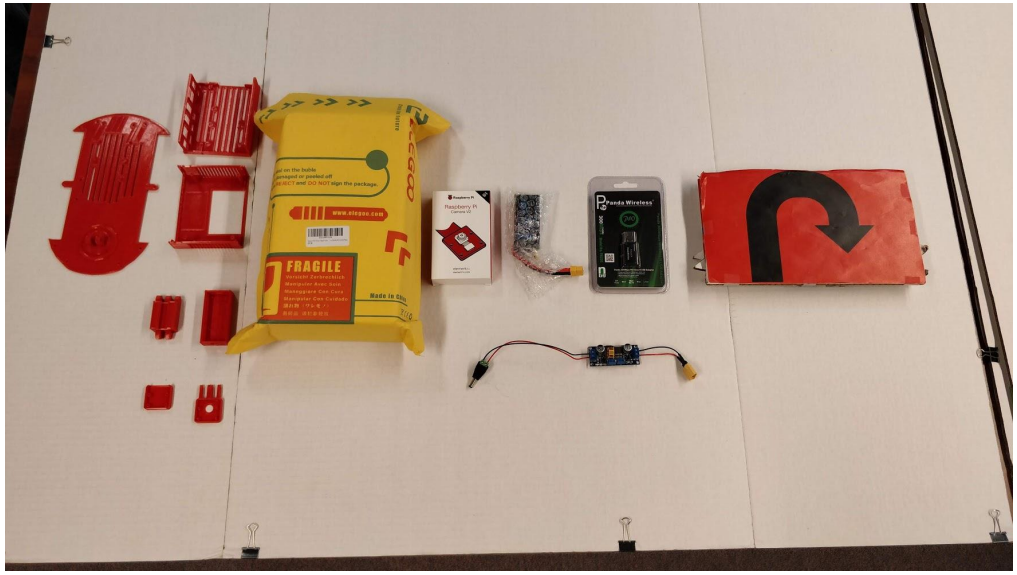
# Topics

- ✓ Overview
- ✓ Engineering
  - ✓ Building RC Car
  - ✓ Wheels Turn
- ✓ Software
  - ✓ Ubuntu
  - ✓ OpenCV
  - ✓ ImageAI
  - ✓ TensorFlow
  - ✓ Program
- ✓ Theory - How it works
- ✓ Training/Running the vehicle

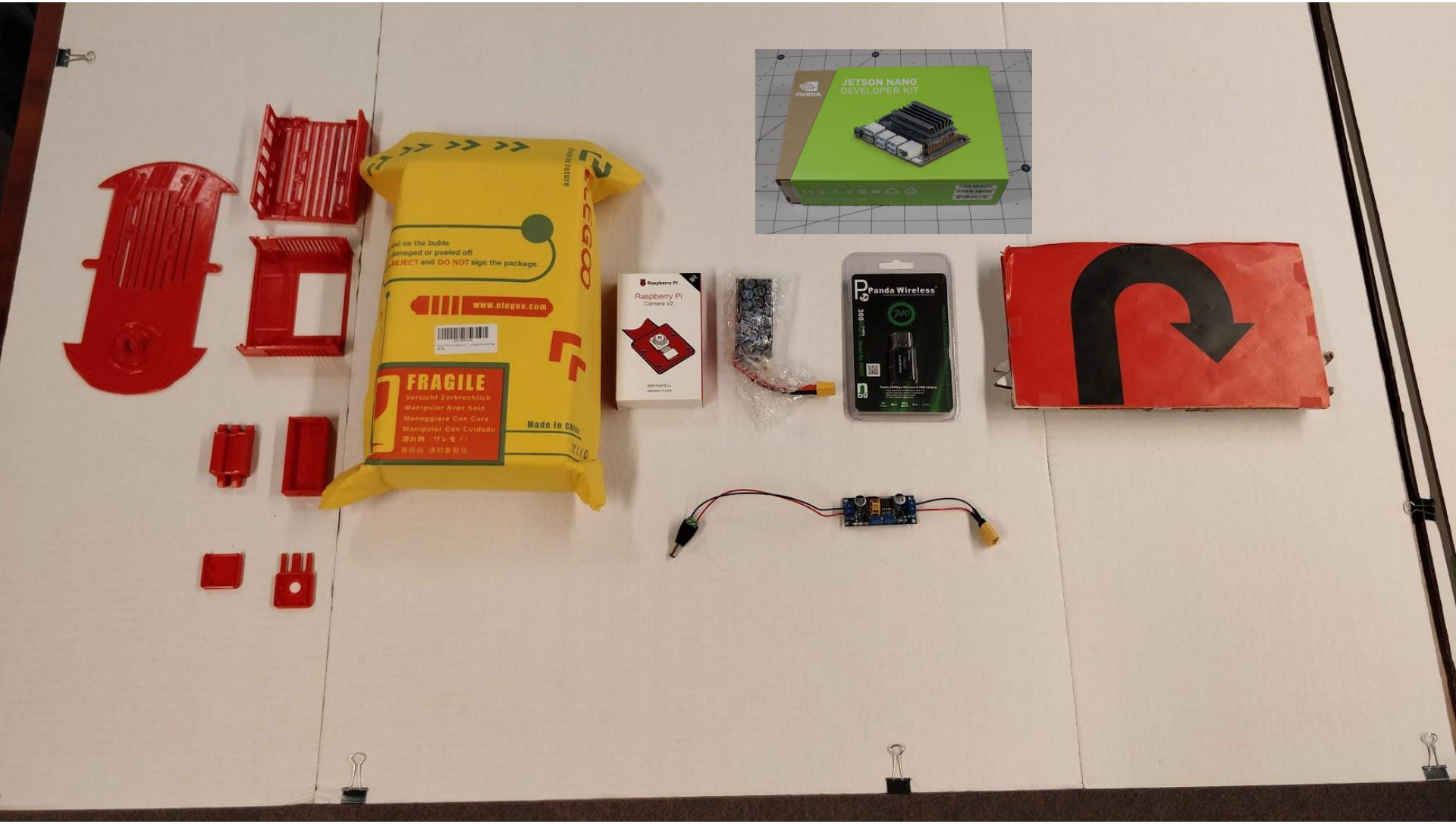
# Building the RC Car

Parts

- 3D printed parts
- Elegoo Smart Robot Car Kit V3.0 Plus
- Jetson Nano
- Raspberry Pi Camera V2
- Lithium Polymer Battery
- Wi-Fi Dongle
- Voltage Regulator Module




# Building the RC Car



# Building the RC Car

# Building the RC Car

# Building the RC Car

# Building the RC Car

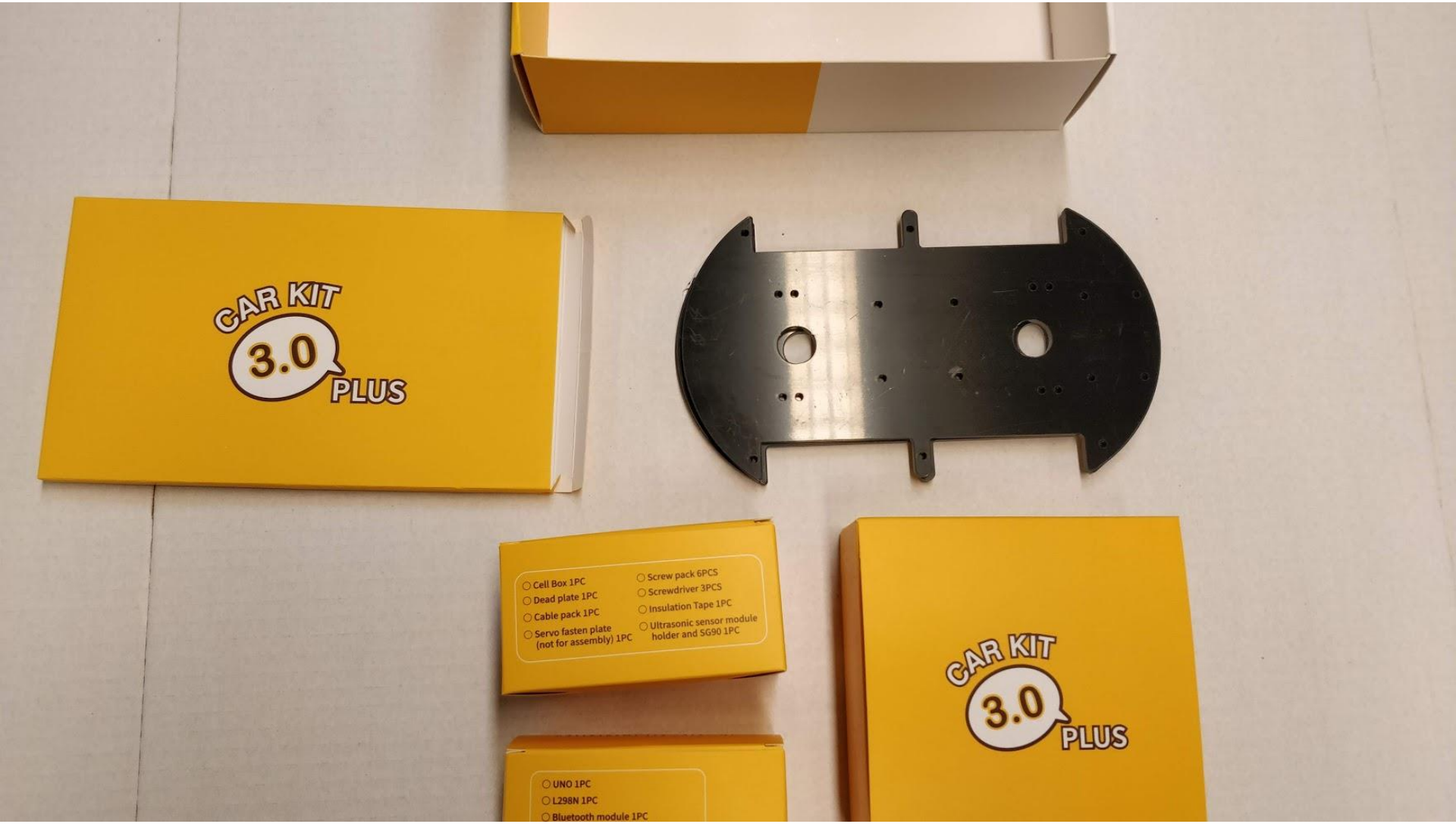
# Building the RC Car

# Building the RC Car

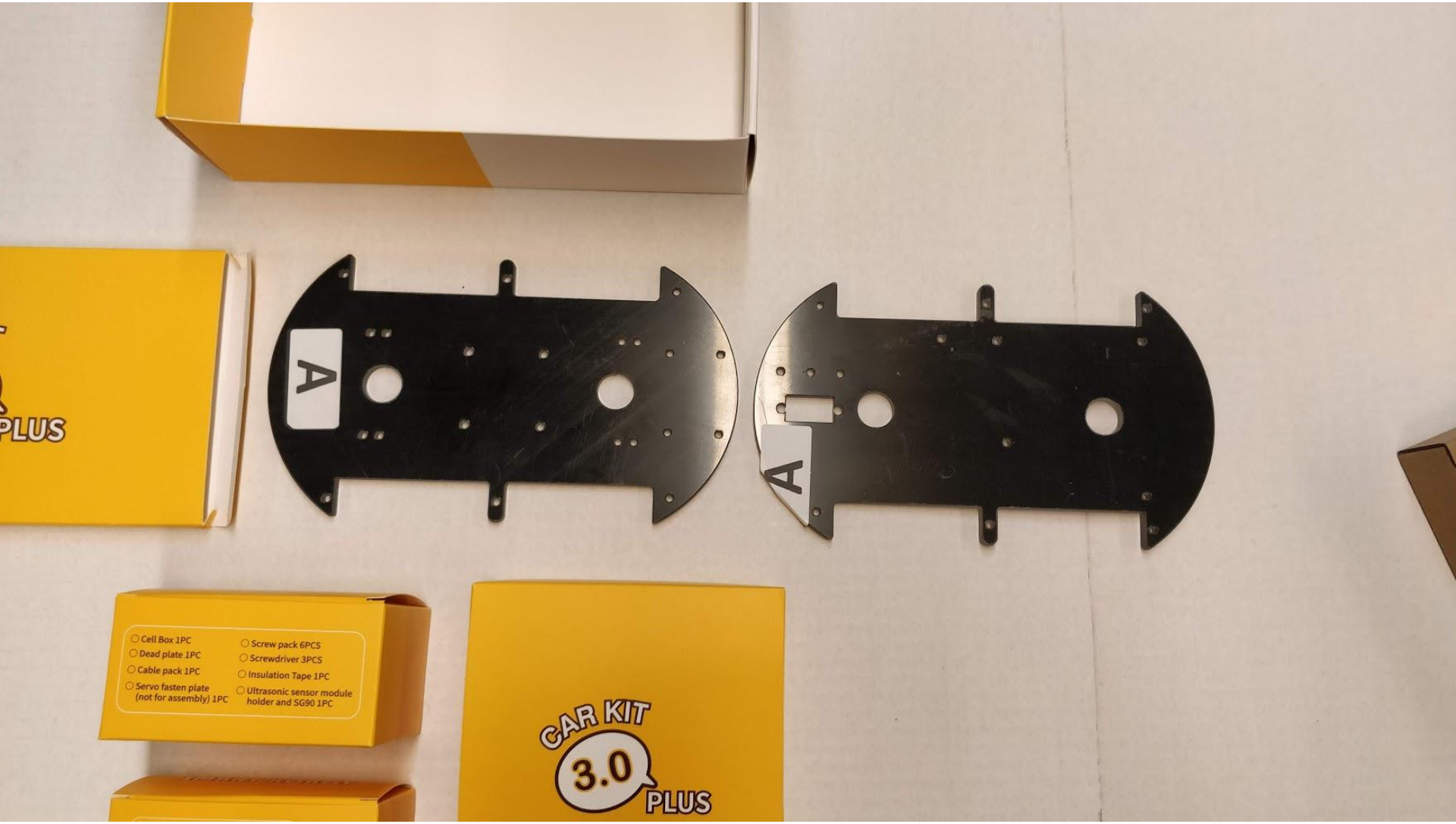
# Building the RC Car

# Building the RC Car

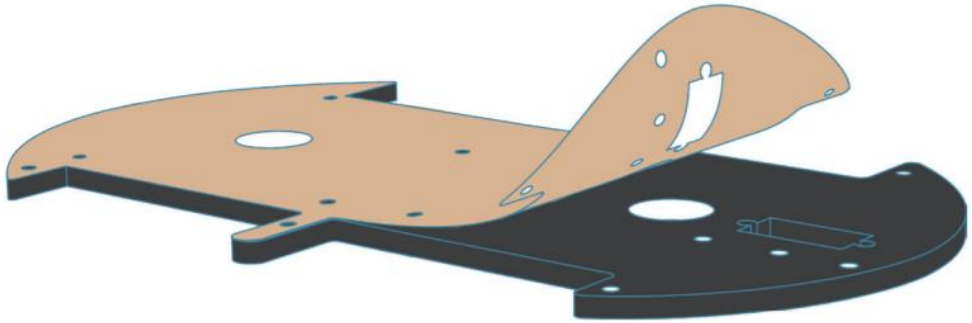
# Building the RC Car



# Building the RC Car

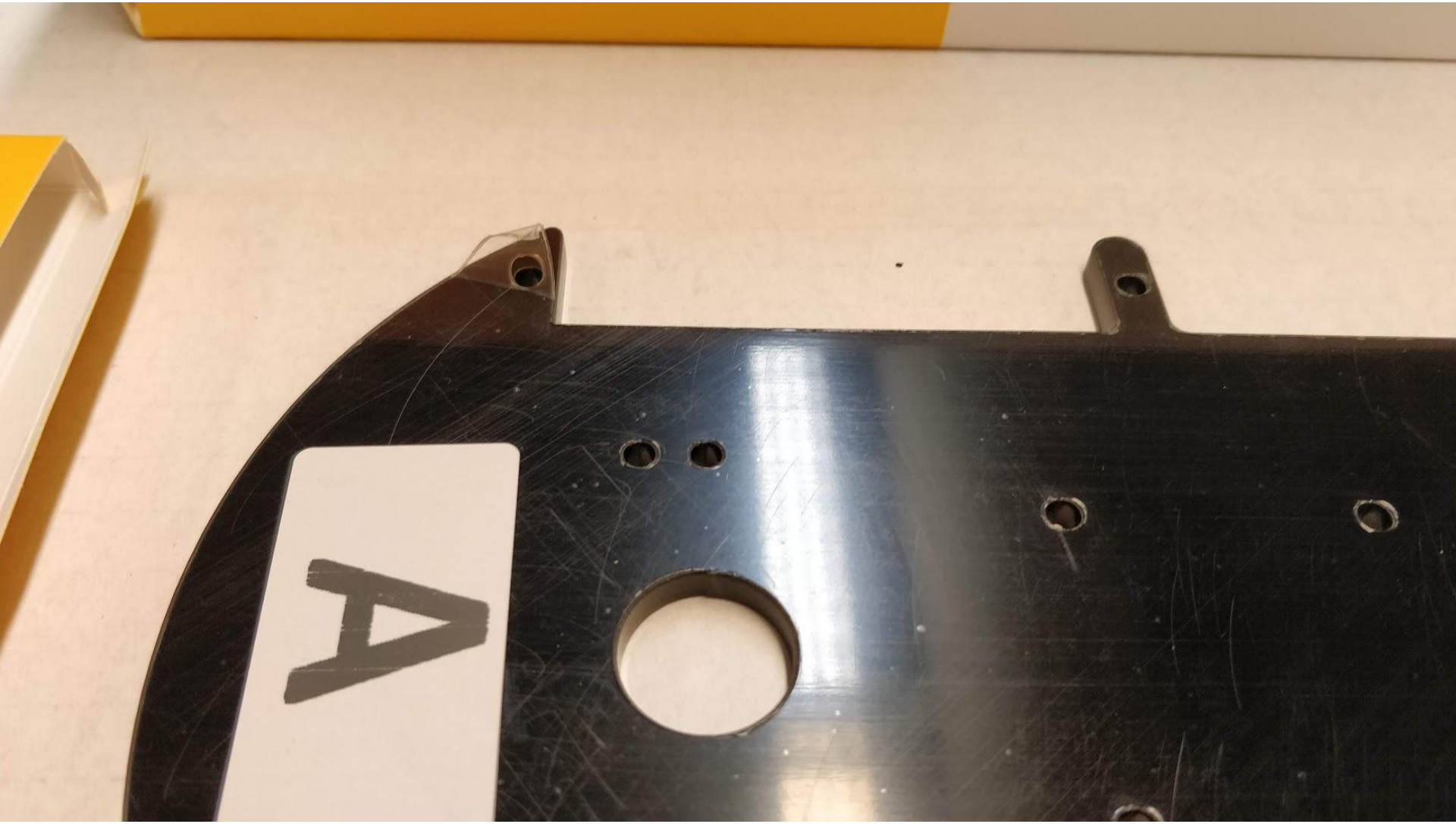


## SMART ROBOT CAR KIT 3.0 PLUS



Attention: Remove the protective film before assembling.

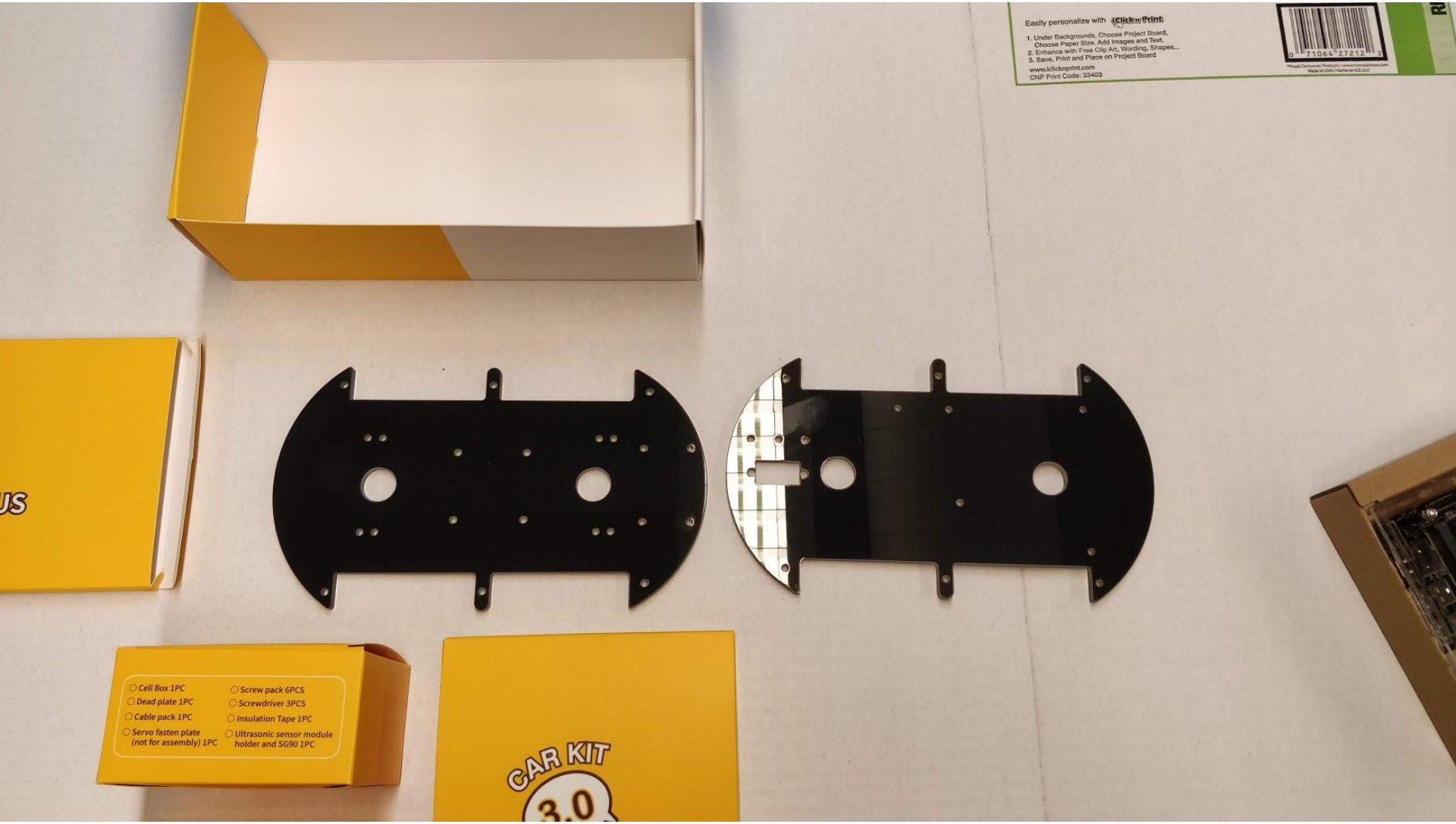
# Building the RC Car



# Building the RC Car

# Building the RC Car

# Building the RC Car

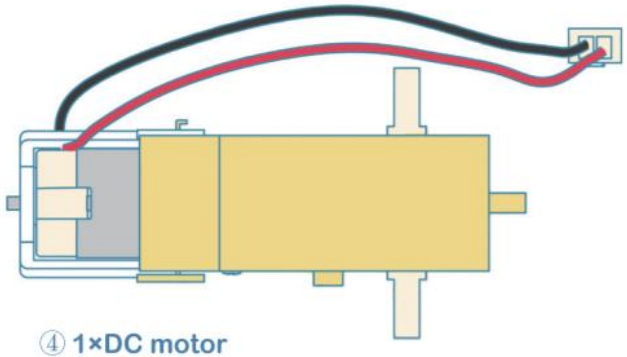


## SMART ROBOT CAR KIT 3.0 PLUS

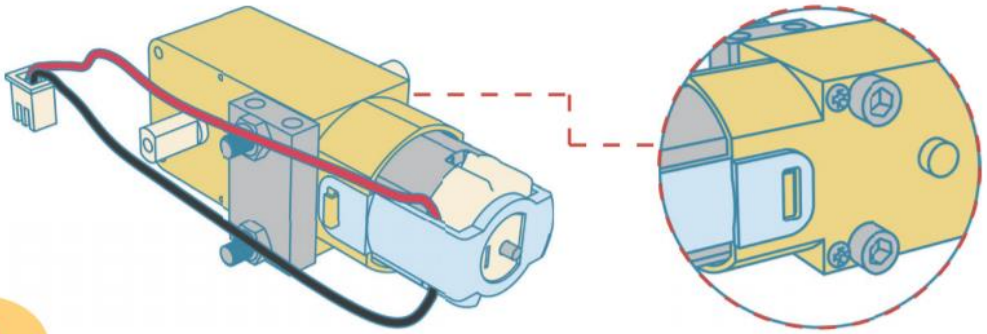
① 2×M3 nuts

② 1×Aluminium Block

③ 2×M3*30 hexagon socket screws



④ 1×DC motor



Take out ① ② ③ from the bag with label "FOR MOTOR".

2

# Building the RC Car

# Building the RC Car

# Building the RC Car


1. FOR MOTOR
M3*30 8PCS
M3*10 8PCS
M3 NUT 8PCS
ALUMINIUM BLOCK 4PCS

# Building the RC Car

# Building the RC Car

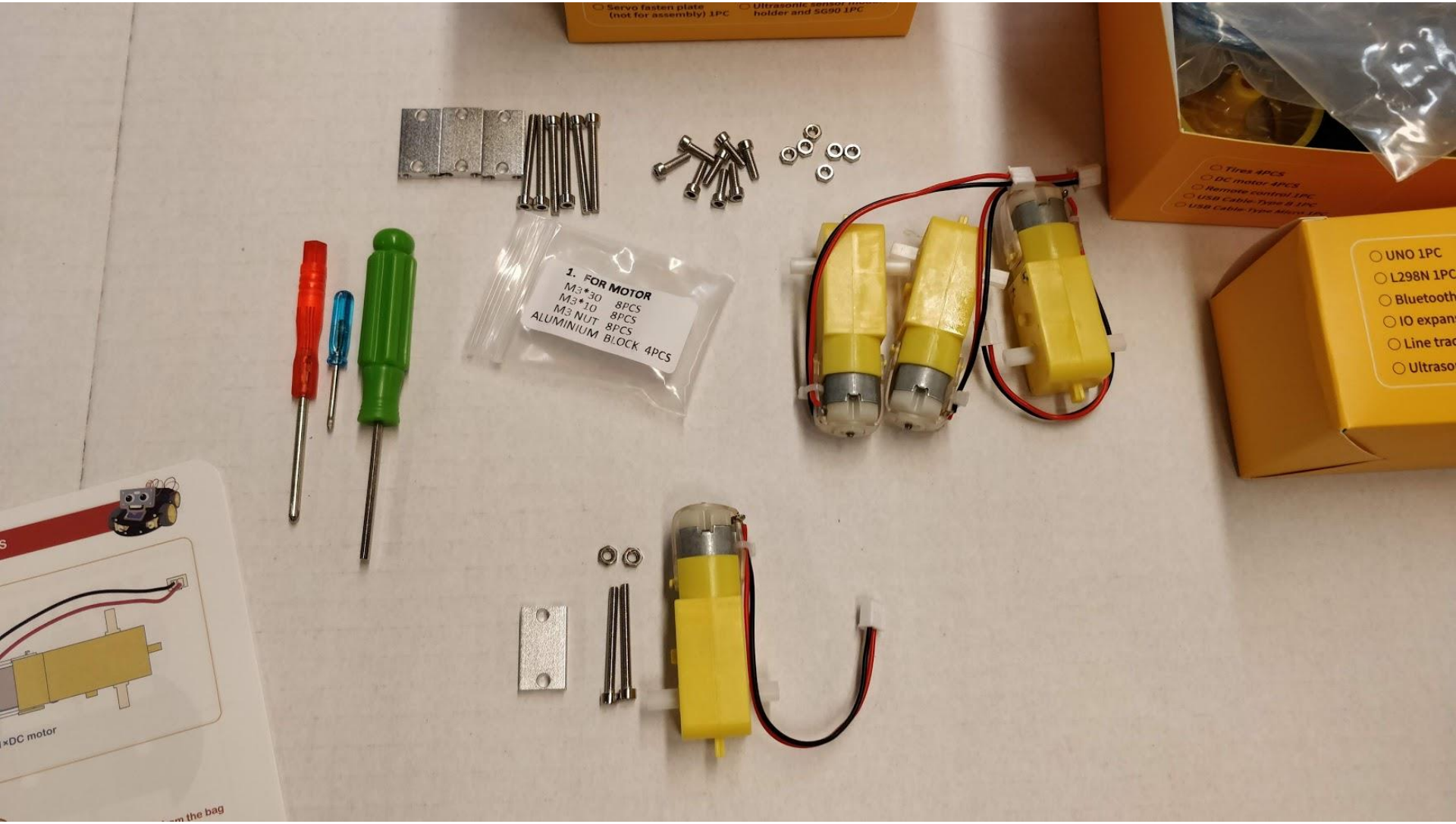
# Building the RC Car

# Building the RC Car

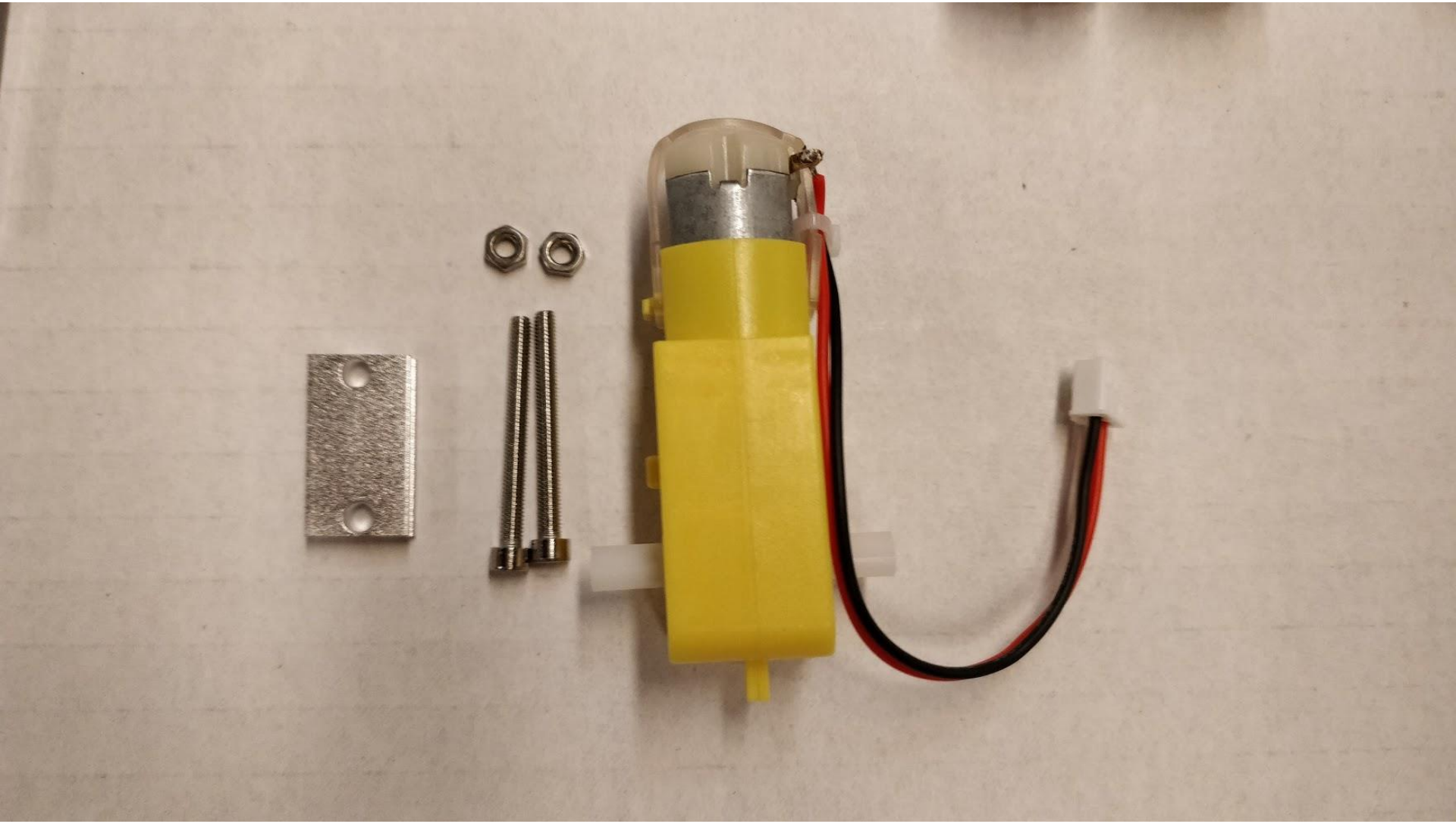
# Building the RC Car

# Building the RC Car

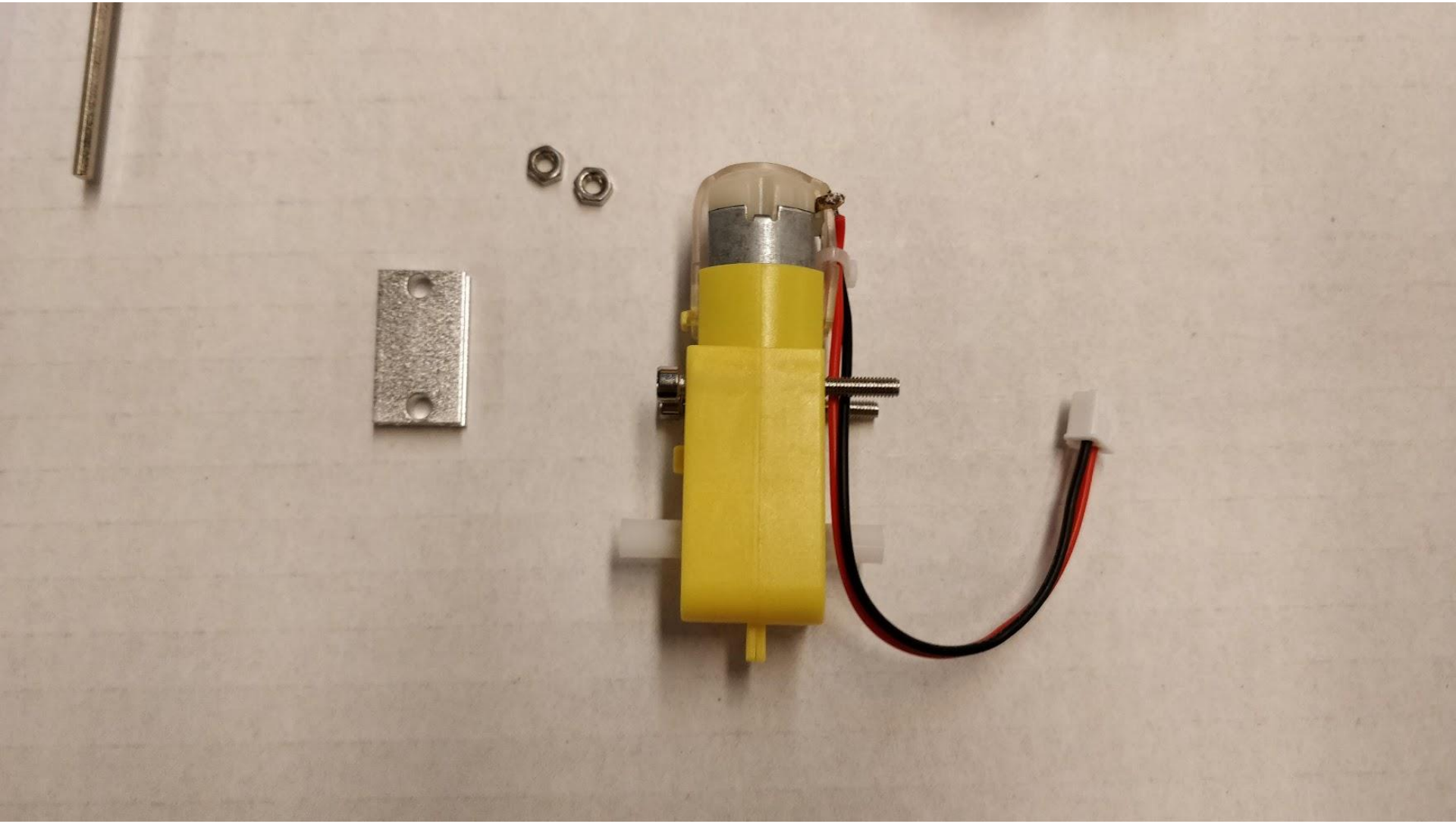
# Building the RC Car



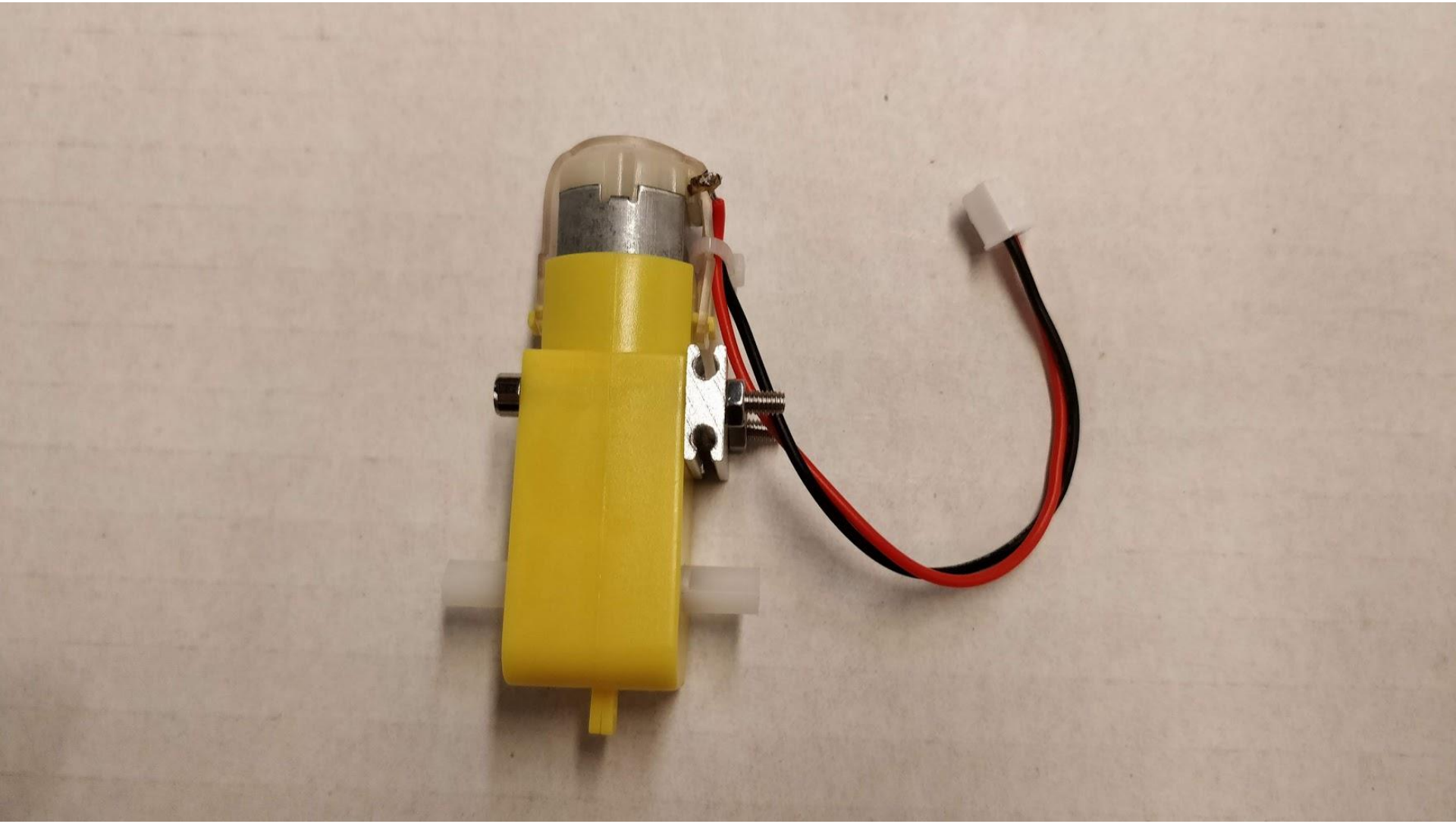
# Building the RC Car



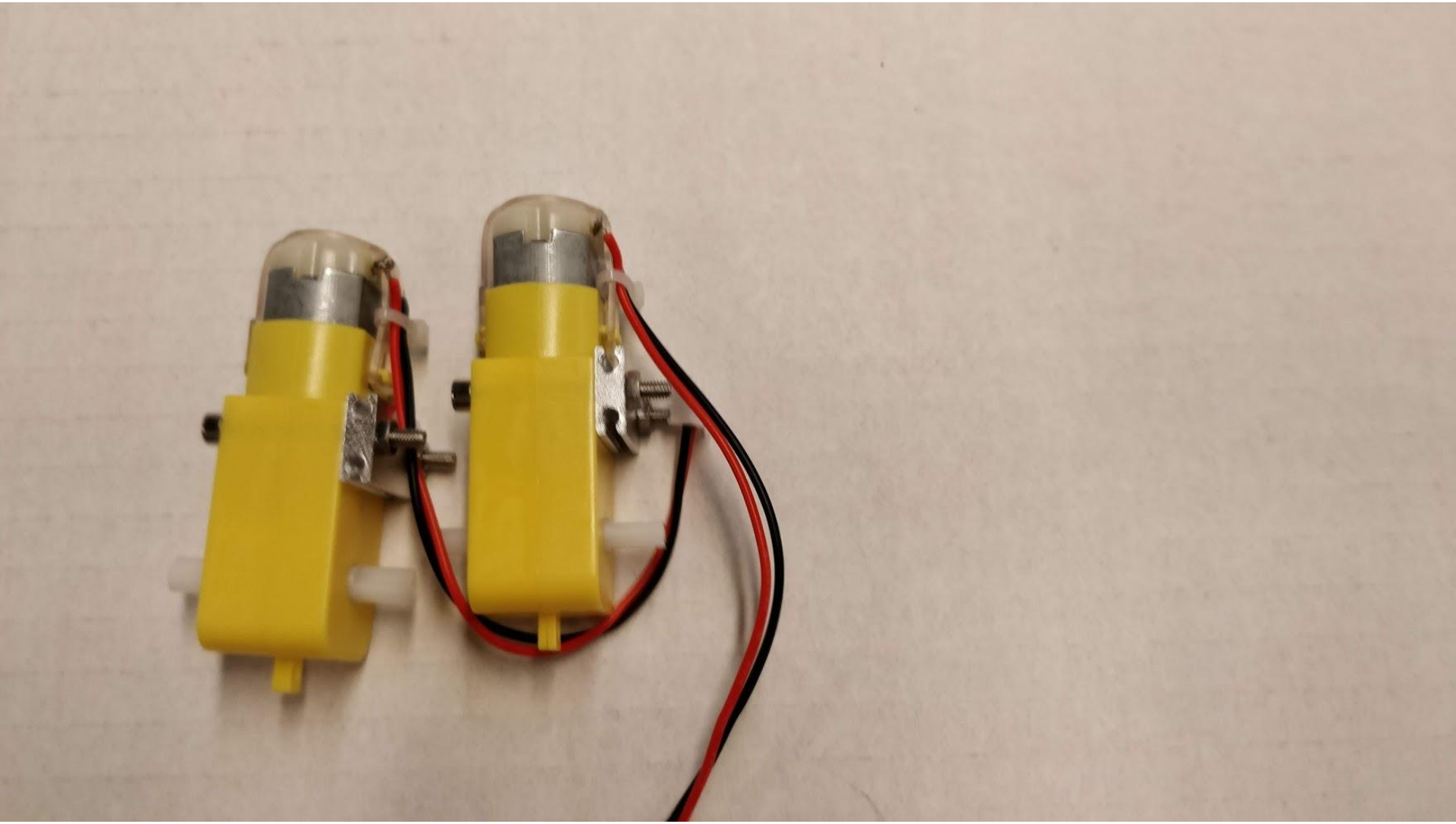
# Building the RC Car



# Building the RC Car

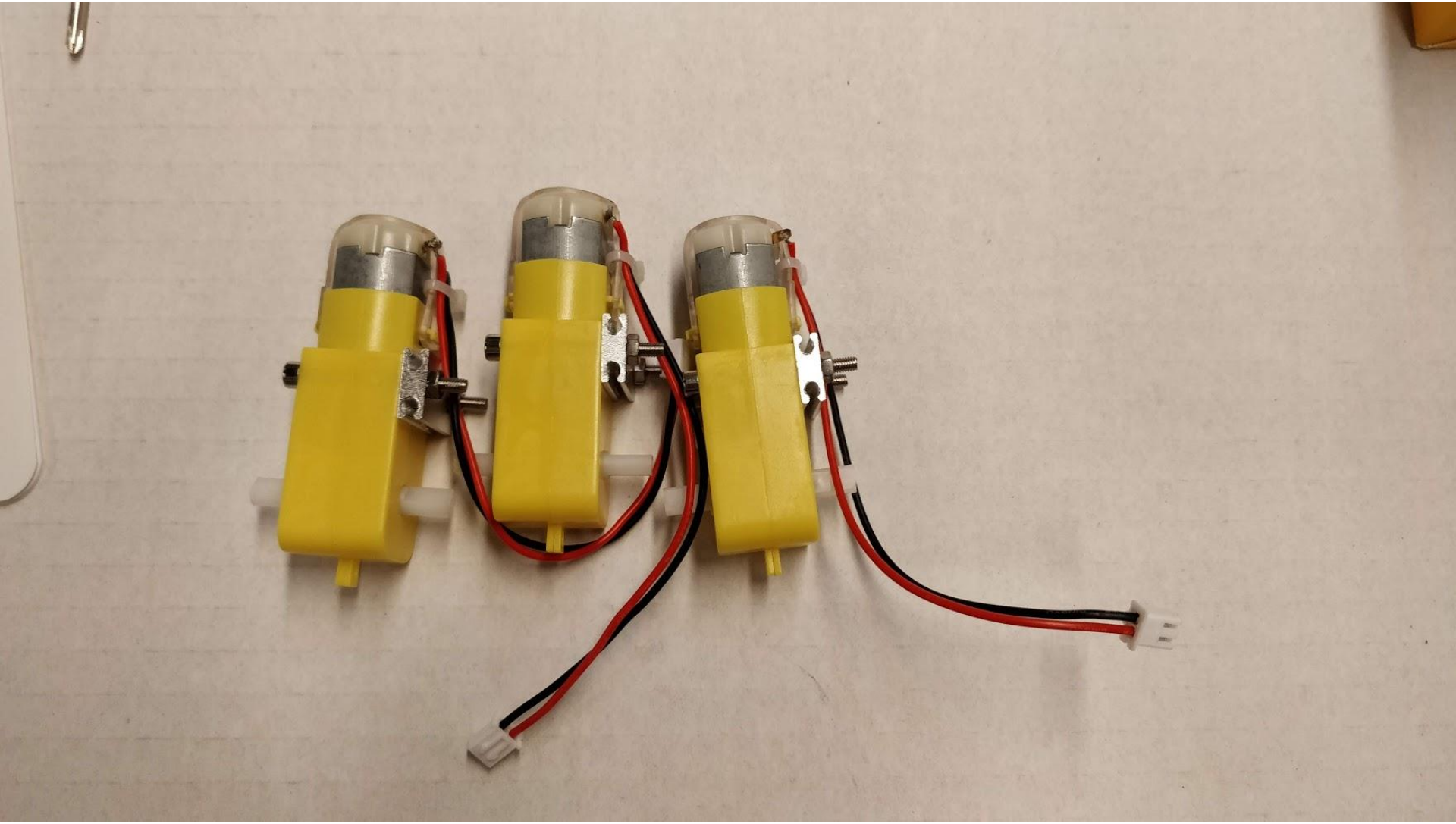


# Building the RC Car

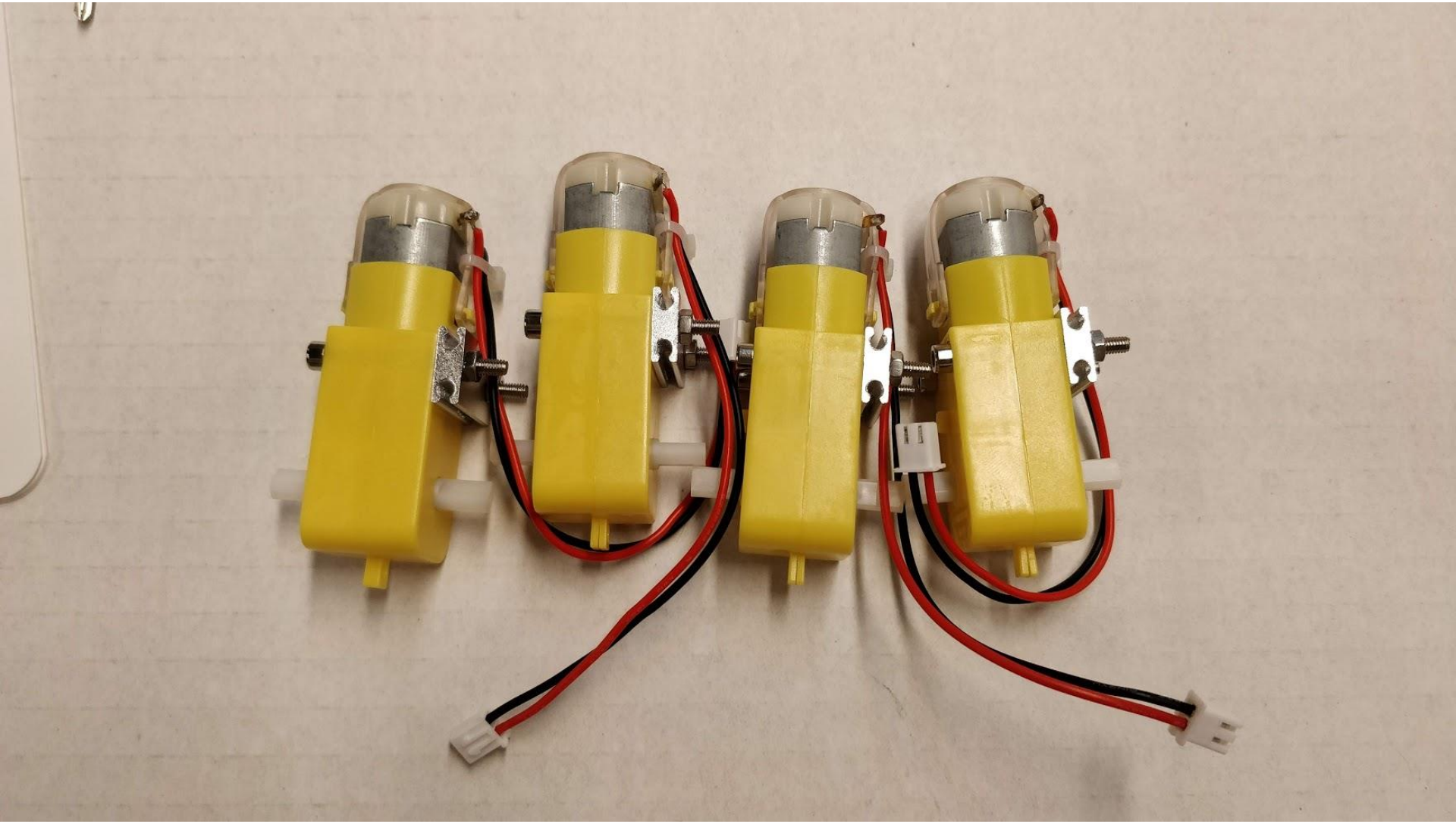


# Building the RC Car



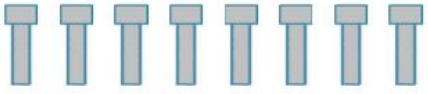


# Building the RC Car

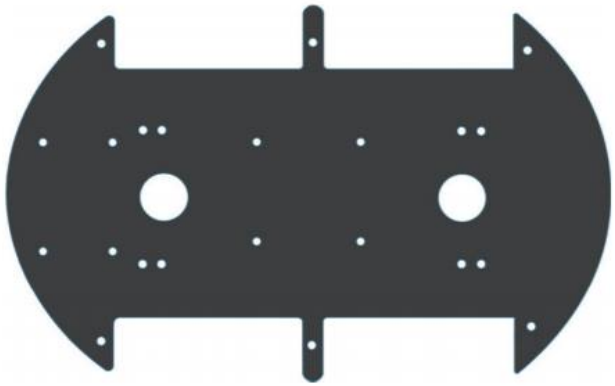


# Building the RC Car

## SMART ROBOT CAR KIT 3.0 PLUS

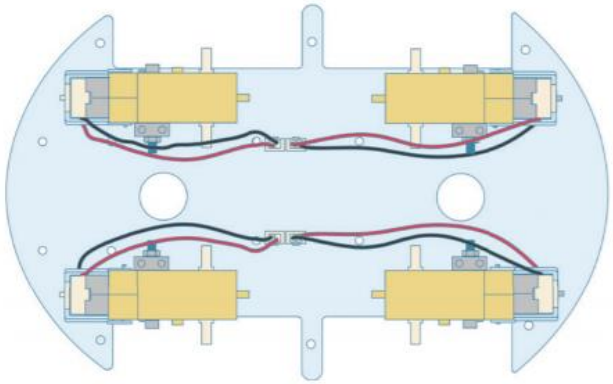


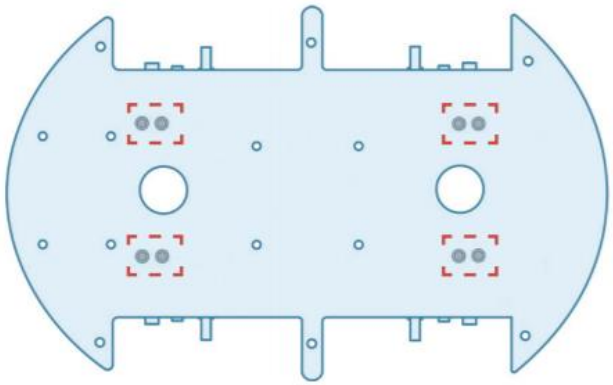
① 8×M3*10 hexagon socket screws



② 1×Acrylic Chassis

Take out ① from the bag with label "FOR MOTOR".



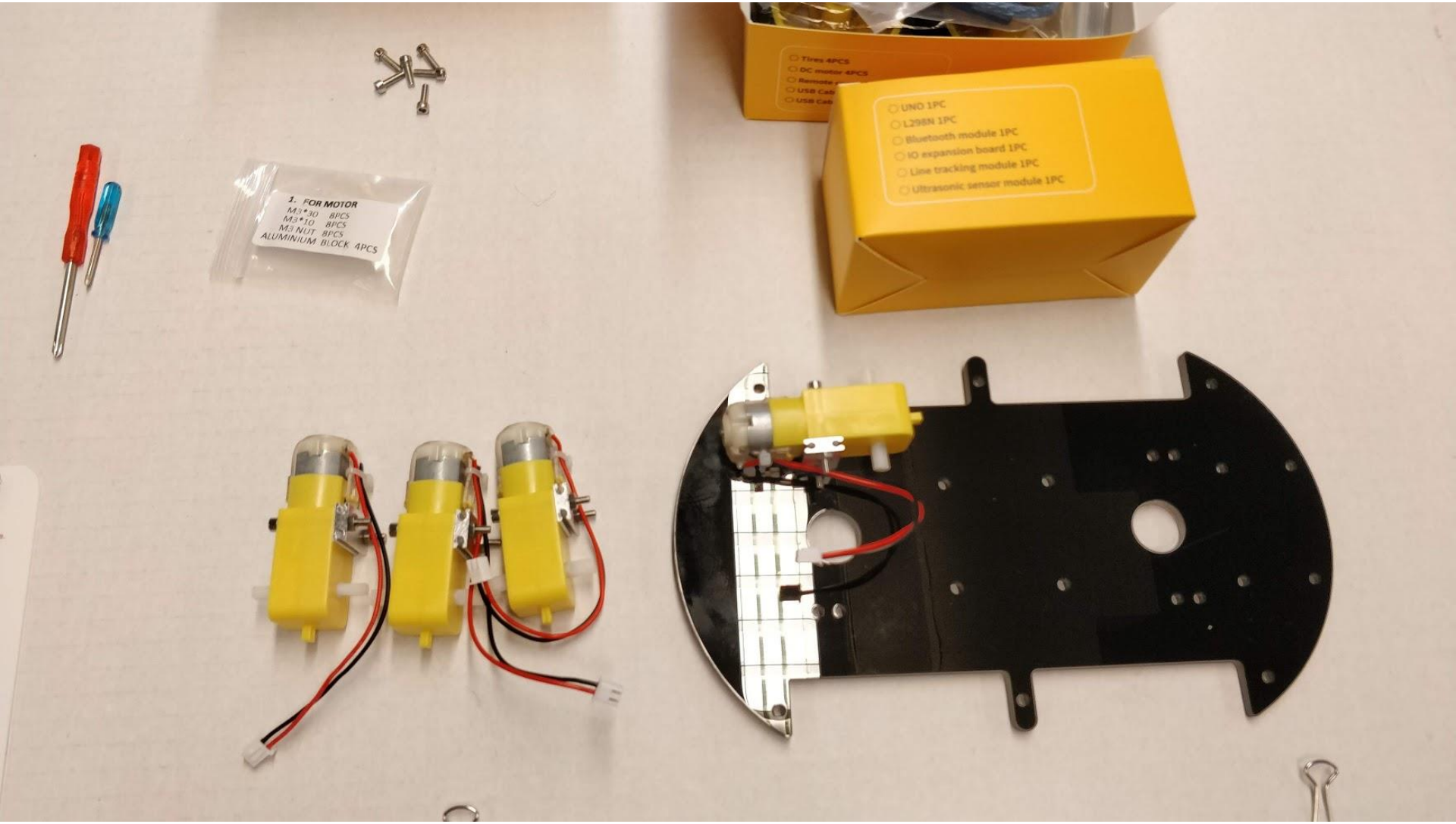


Back side

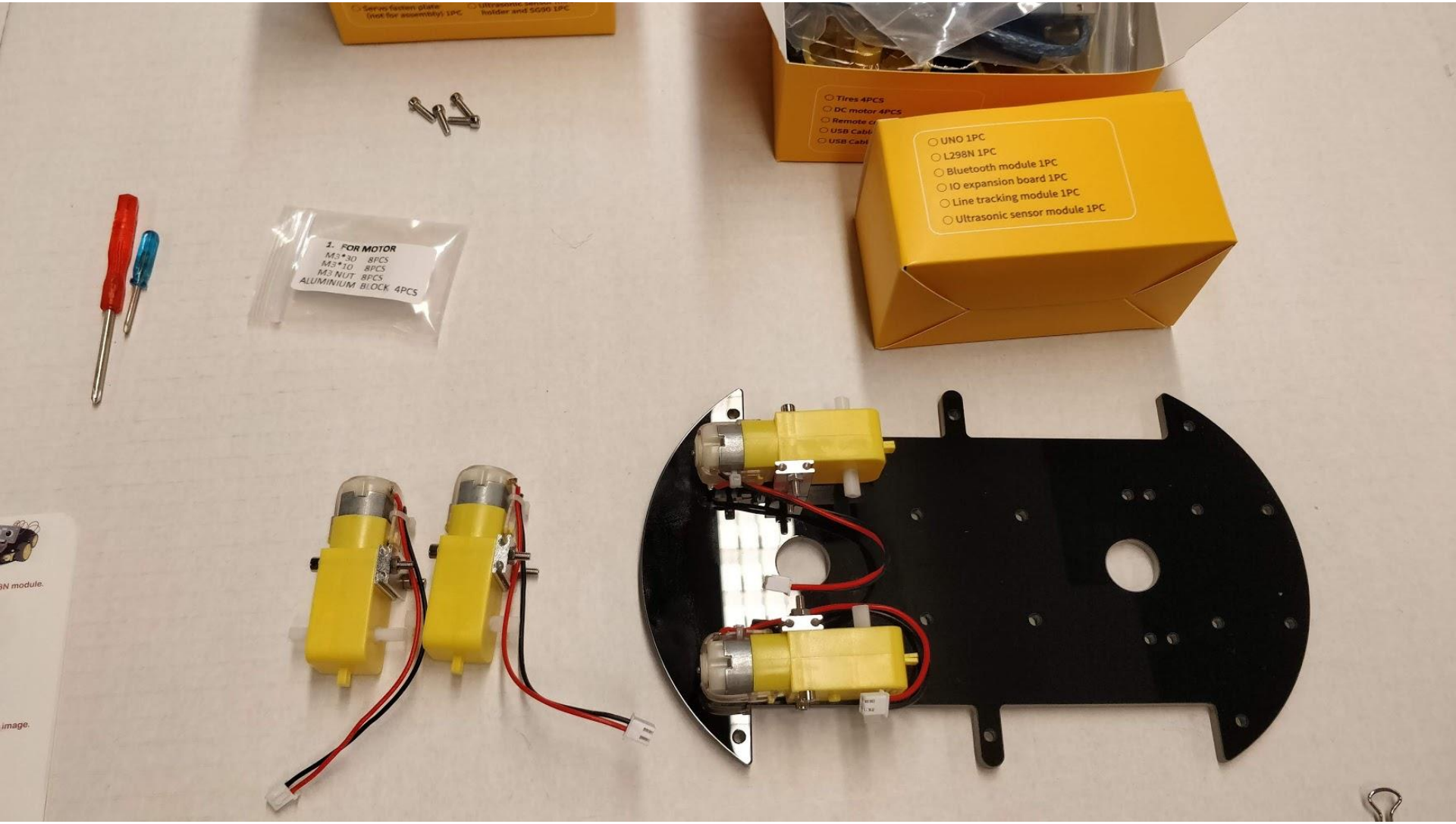
# Building the RC Car

# Building the RC Car

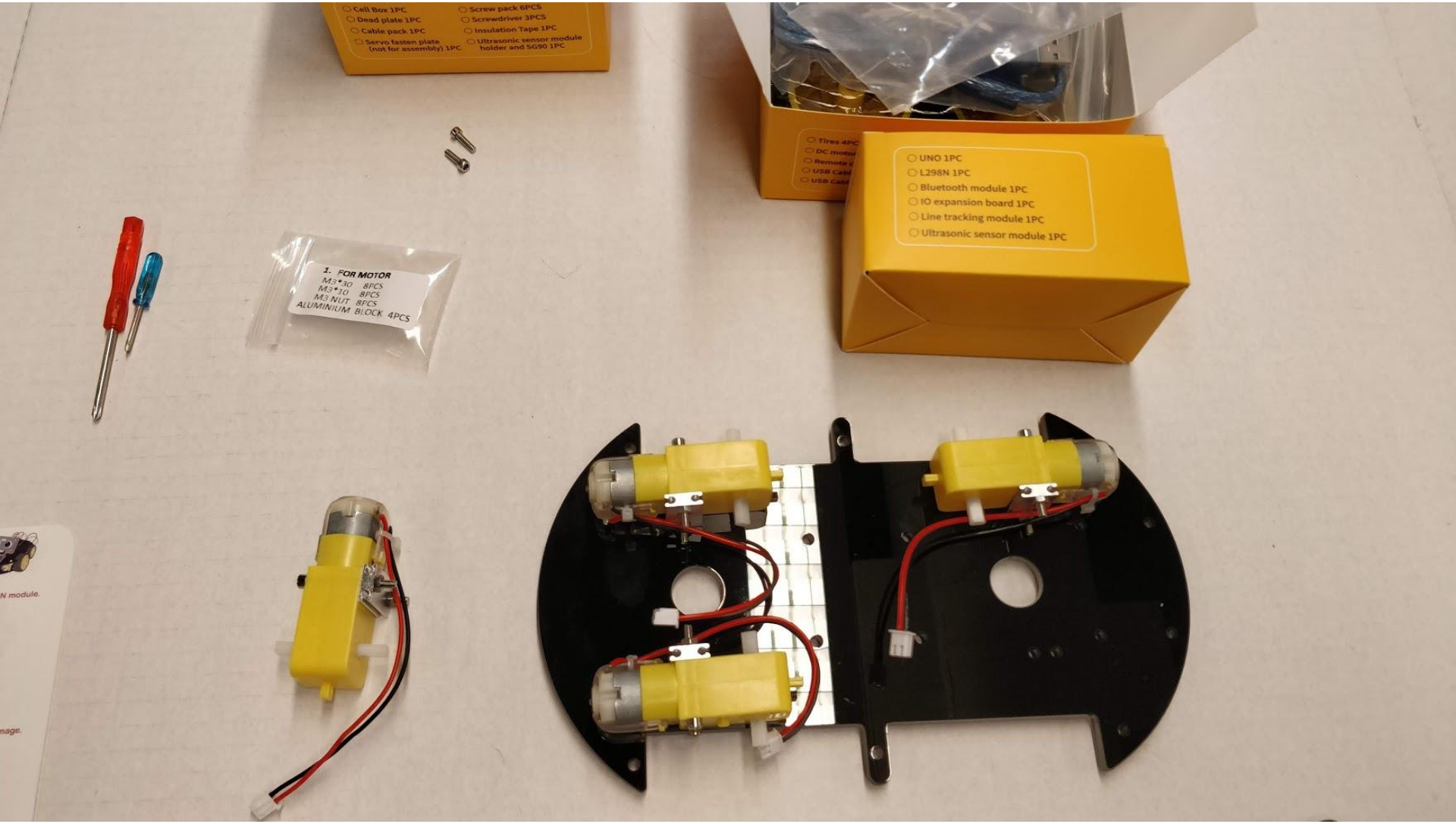
# Building the RC Car



# Building the RC Car



# Building the RC Car



# Building the RC Car

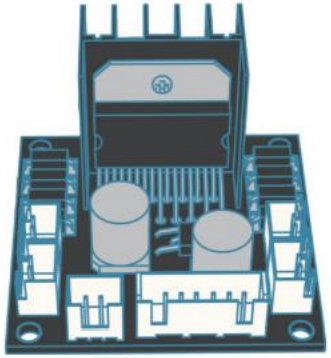
## SMART ROBOT CAR KIT 3.0 PLUS

① 4×M3*14 hexagon socket screws

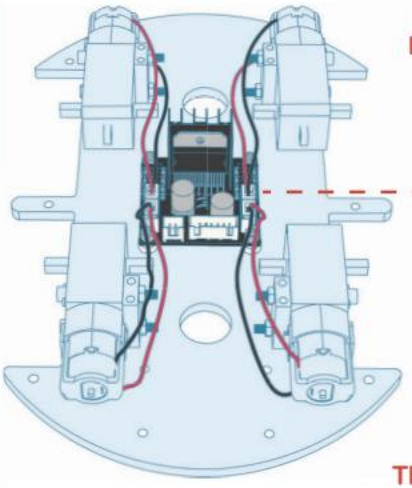
② 4×M3 nuts

③ 4×separation shim

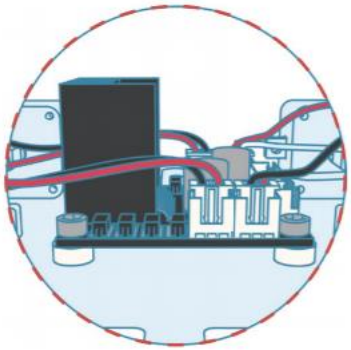
Take out ① ② ③ from the bag with label "FOR UNO、L298N".



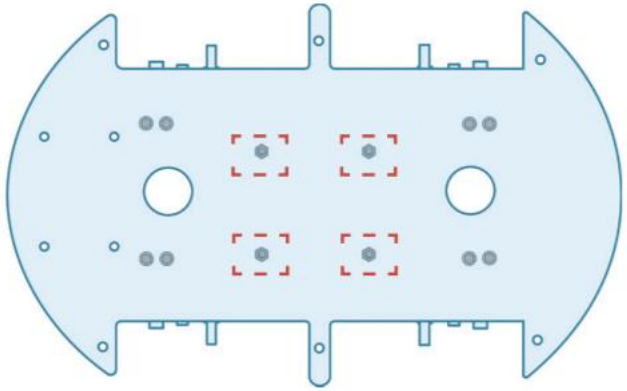
④ 1×L298N



Pay attention to the direction of L298N module.



The wiring of the DC motor is shown in the image.

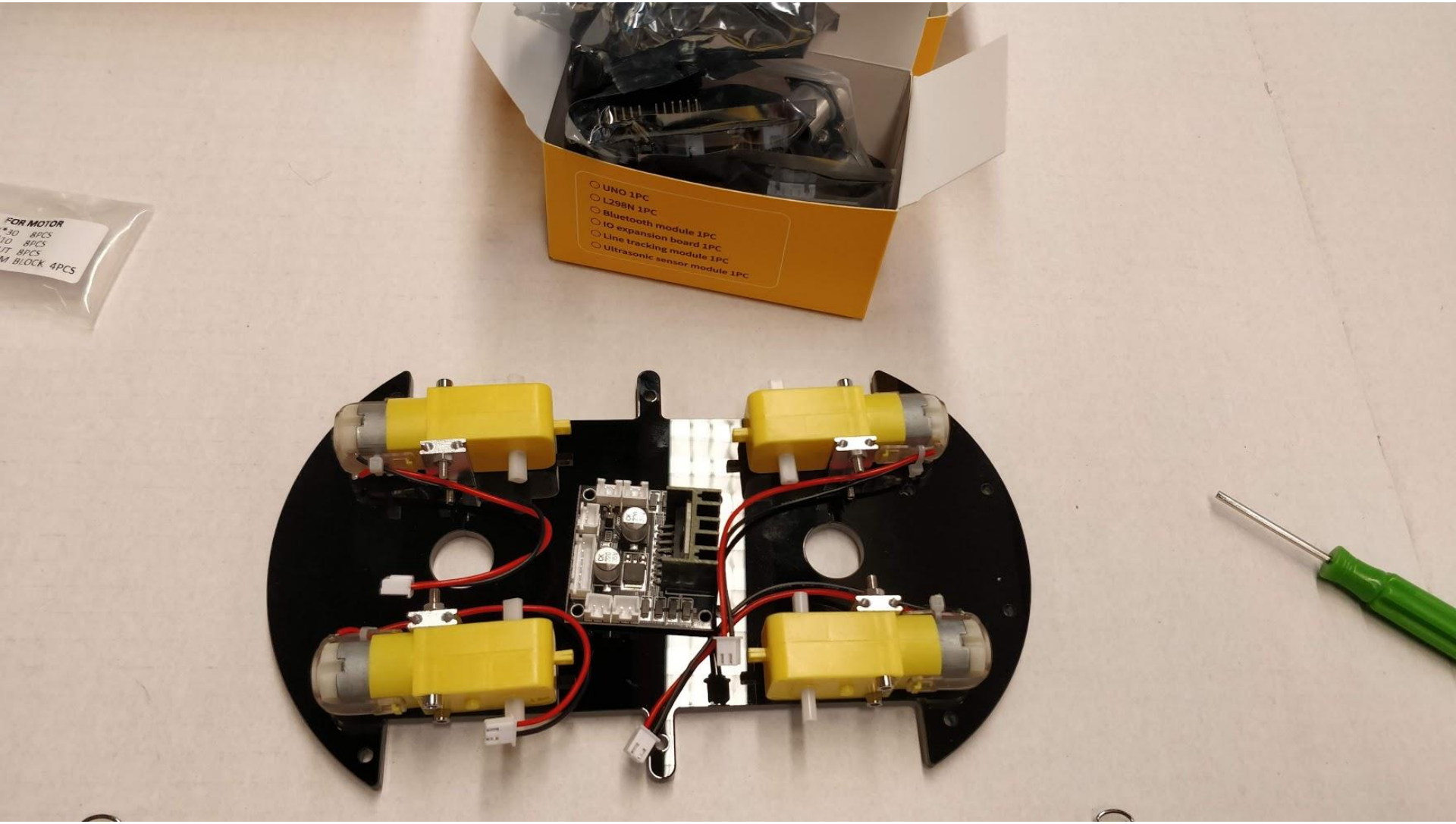


Back side

# Building the RC Car

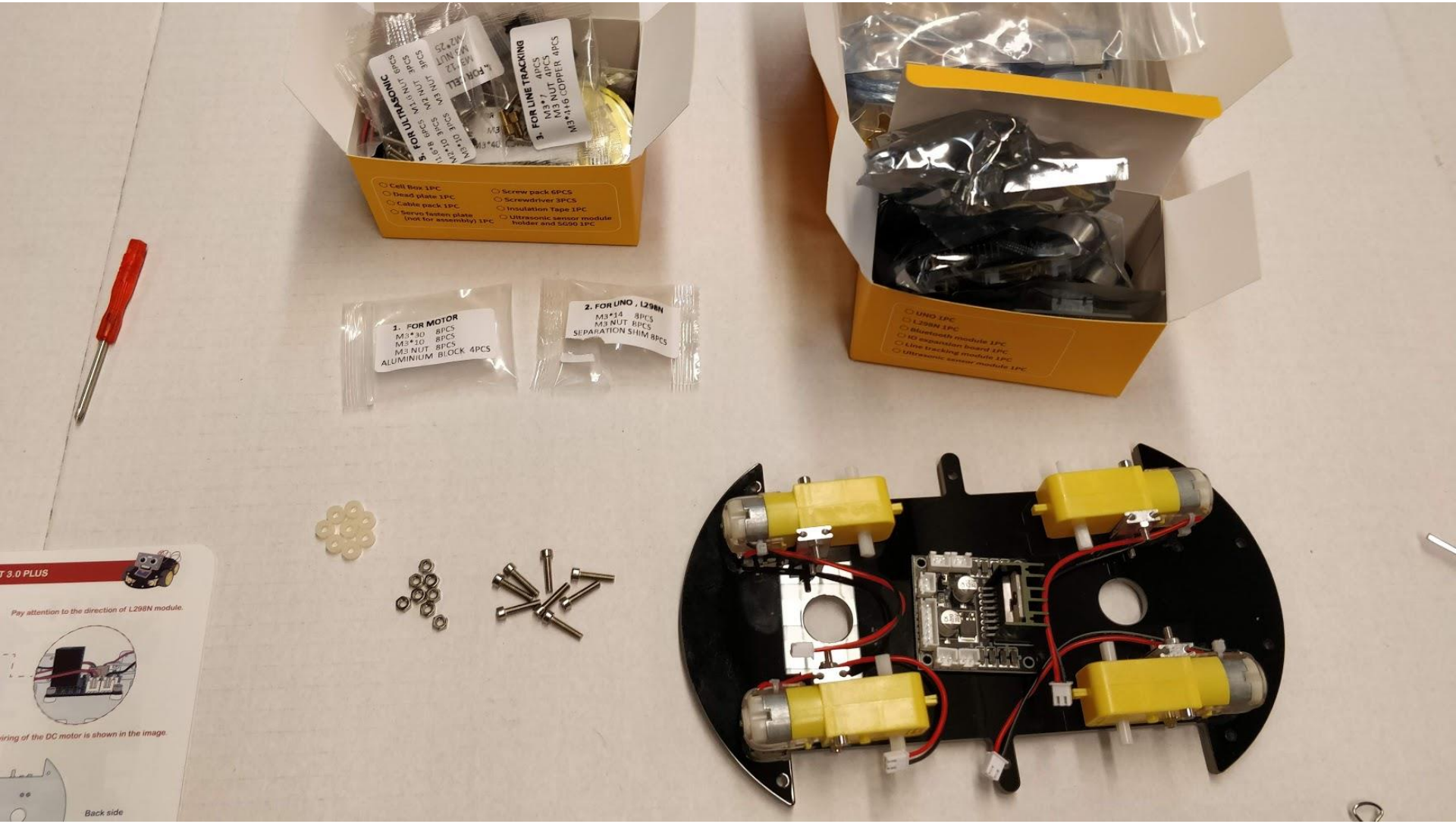
# Building the RC Car

# Building the RC Car

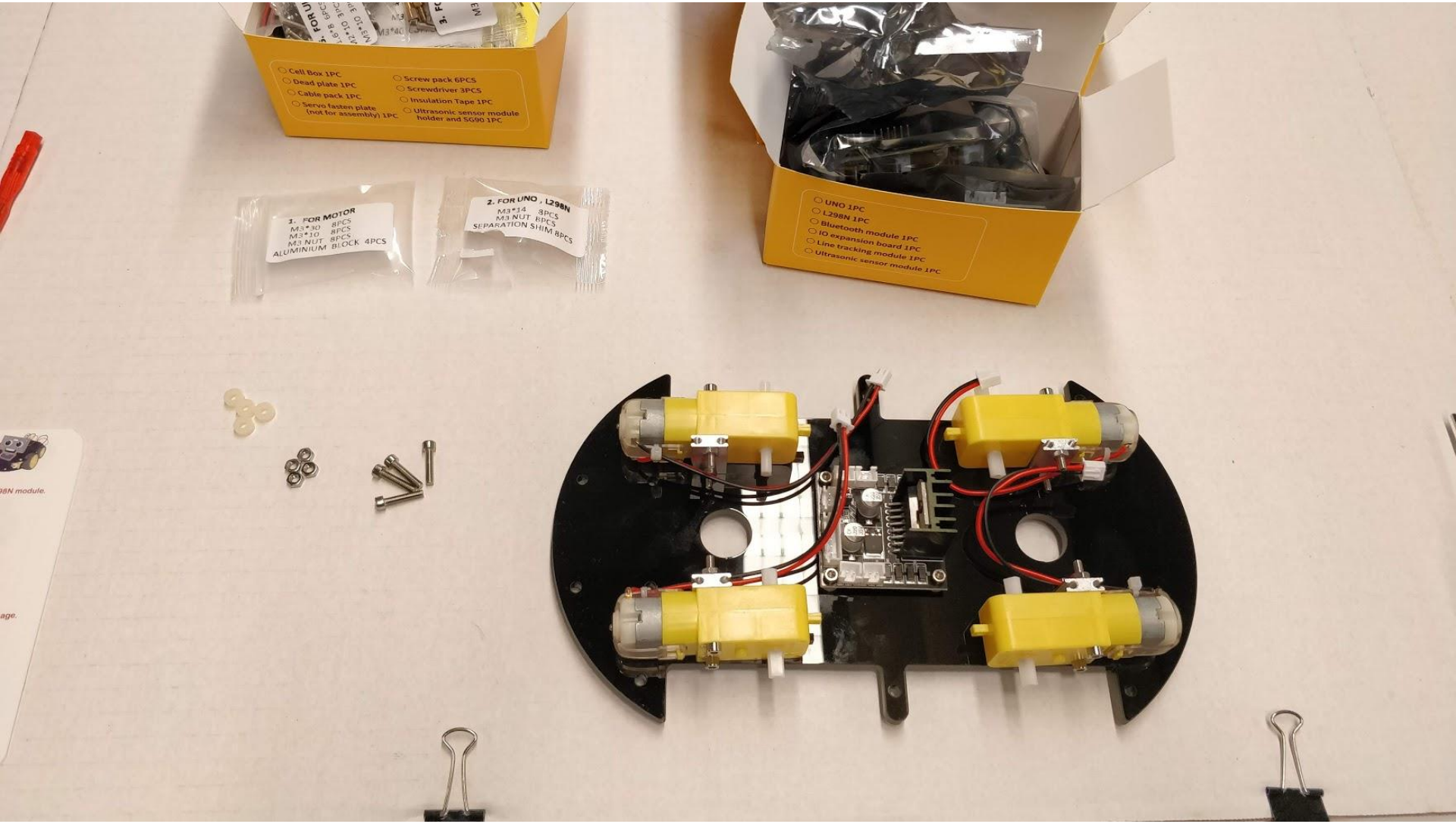


# Building the RC Car

# Building the RC Car



# Building the RC Car



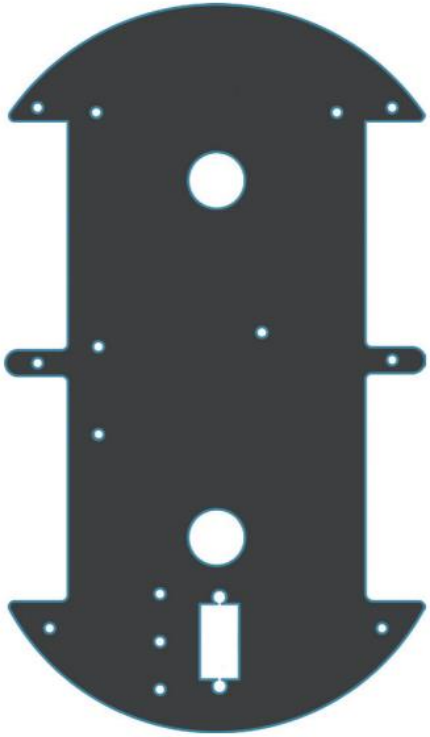
# Building the RC Car

## SMART ROBOT CAR KIT 3.0 PLUS

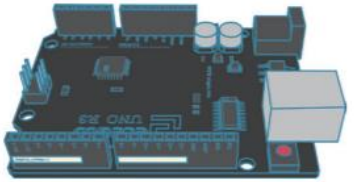
① 3× M3*14 hexagon socket screws

② 3×separation shim

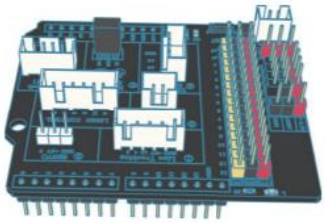
③ 3× M3 nuts



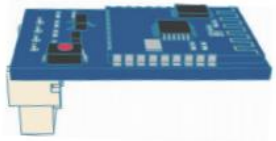
④ 1×Acrylic Chassis



⑤ 1×UNO R3 board



⑥ 1×IO expansion board

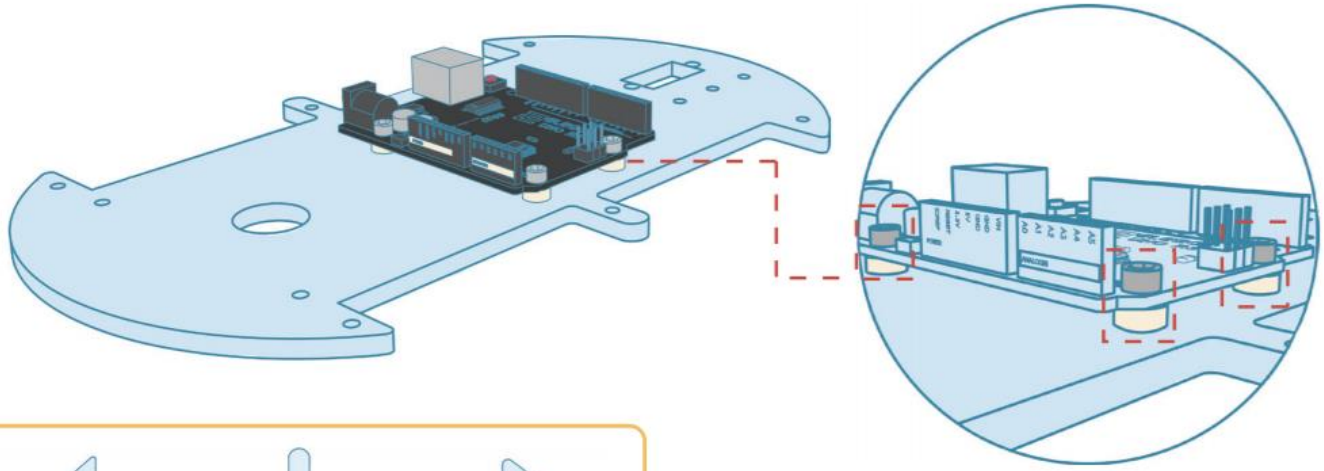


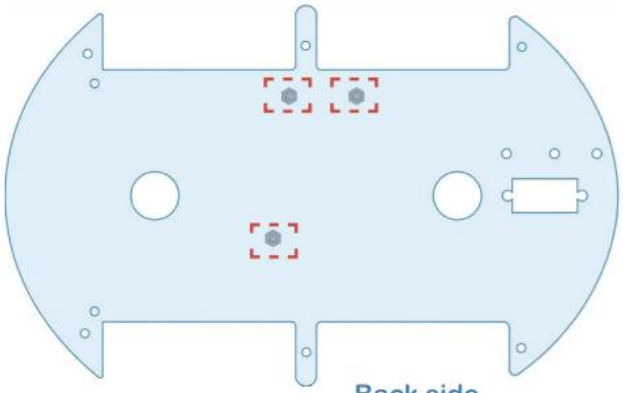
⑦ 1×Bluetooth module

6

Take out ① ② ③ from the bag with label " FOR UNO、L298N "

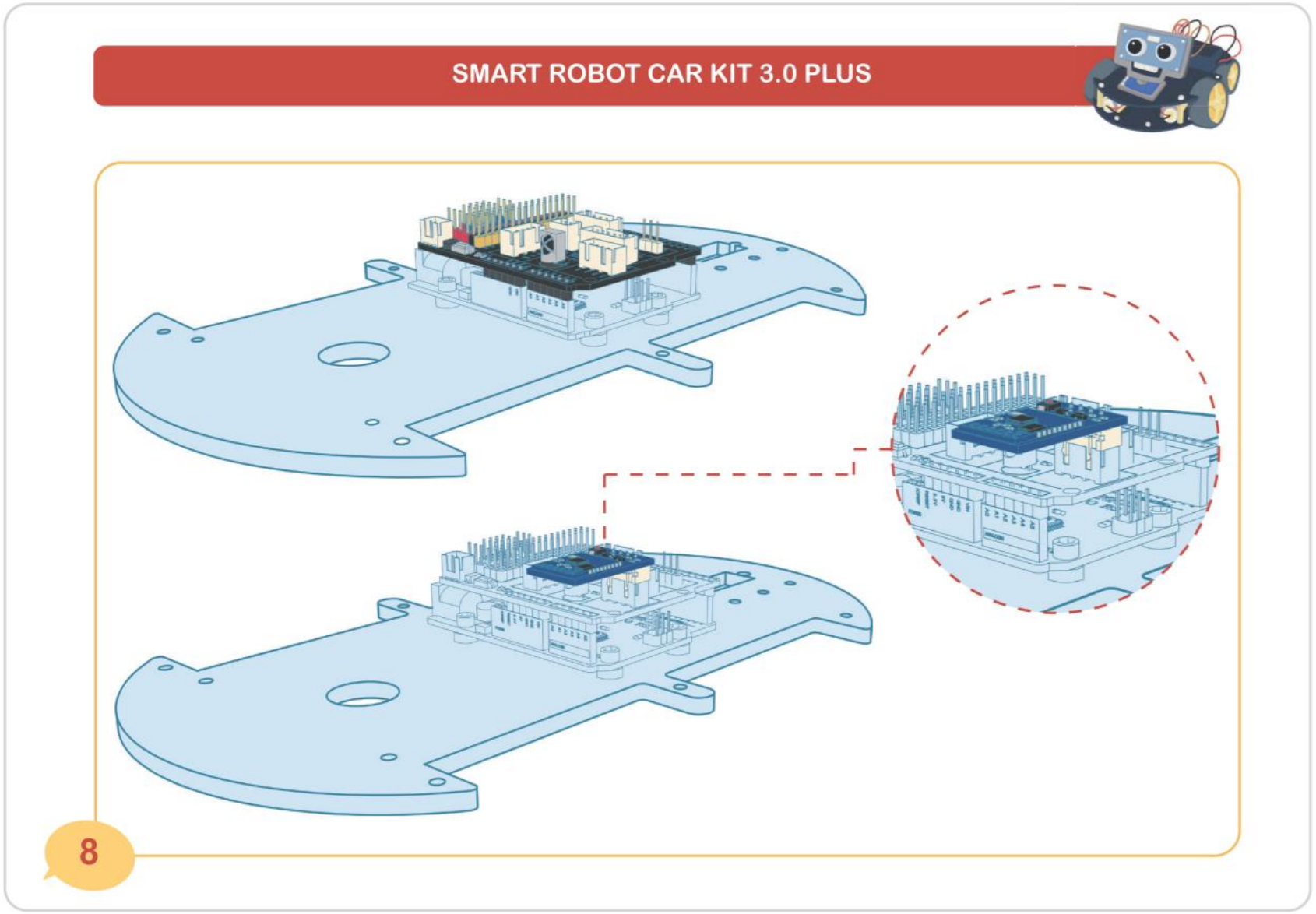
## SMART ROBOT CAR KIT 3.0 PLUS



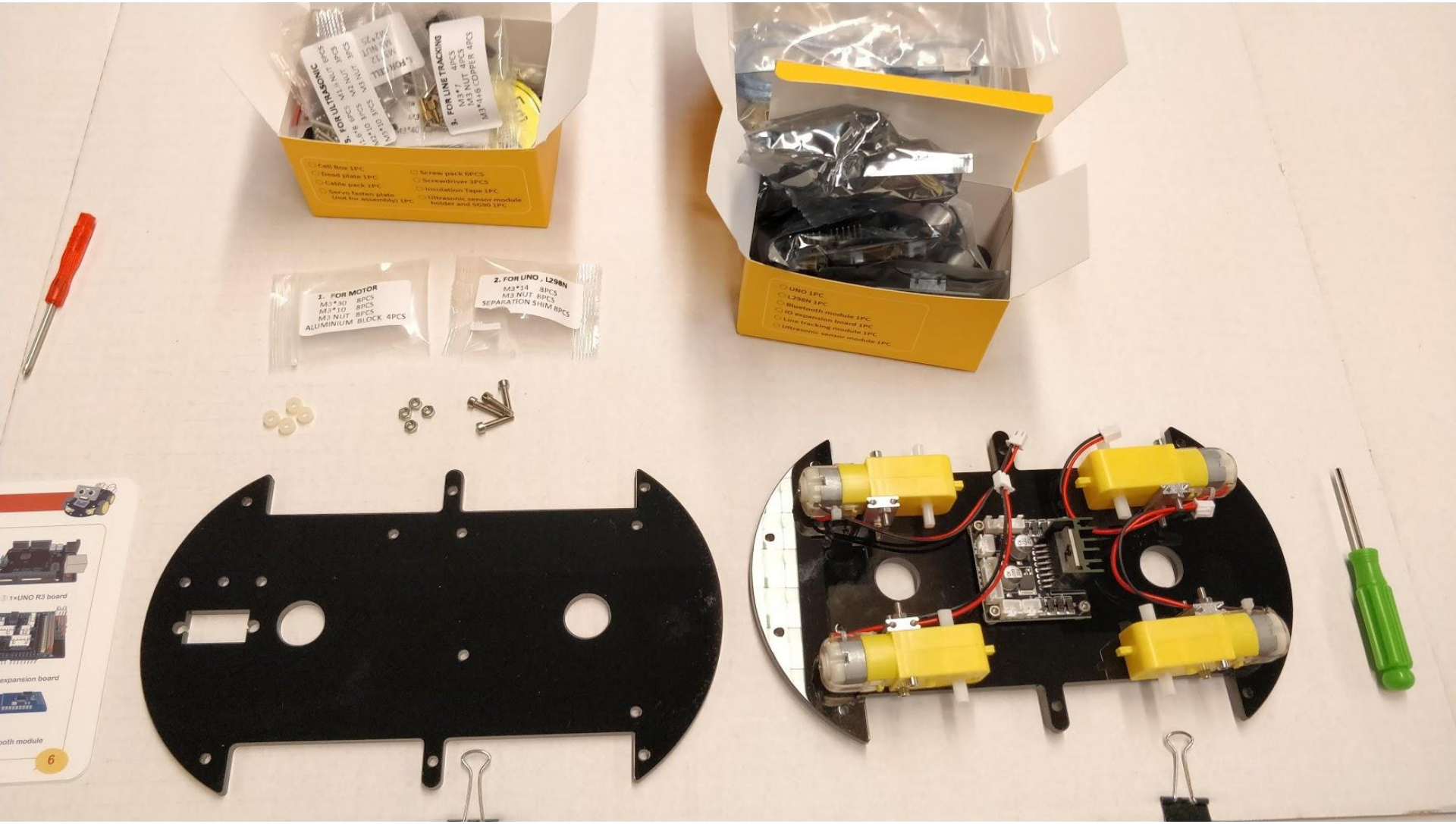



Pay attention to the front and back side of the acrylic chassis when assembling.

# Building the RC Car

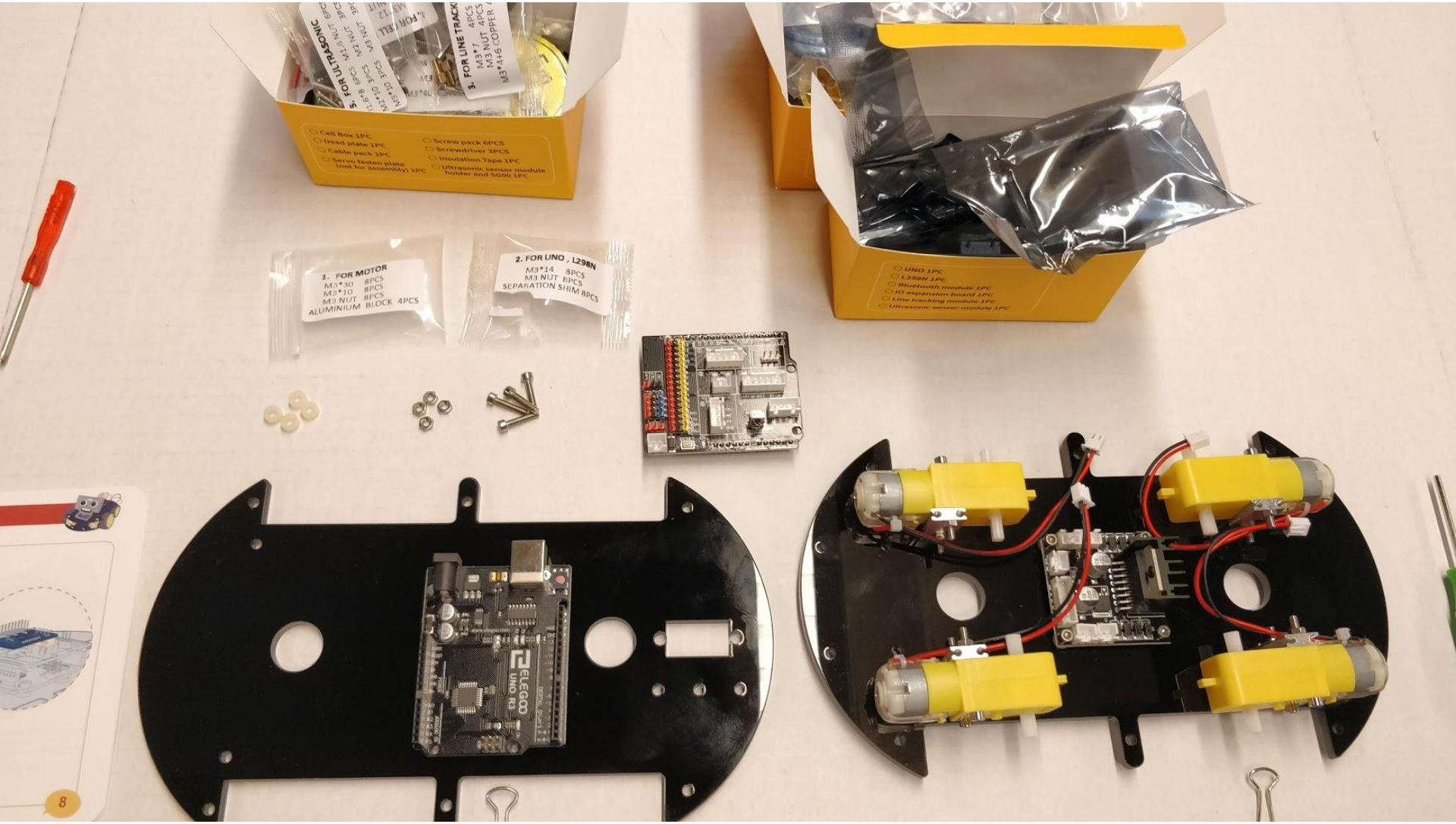


# Building the RC Car

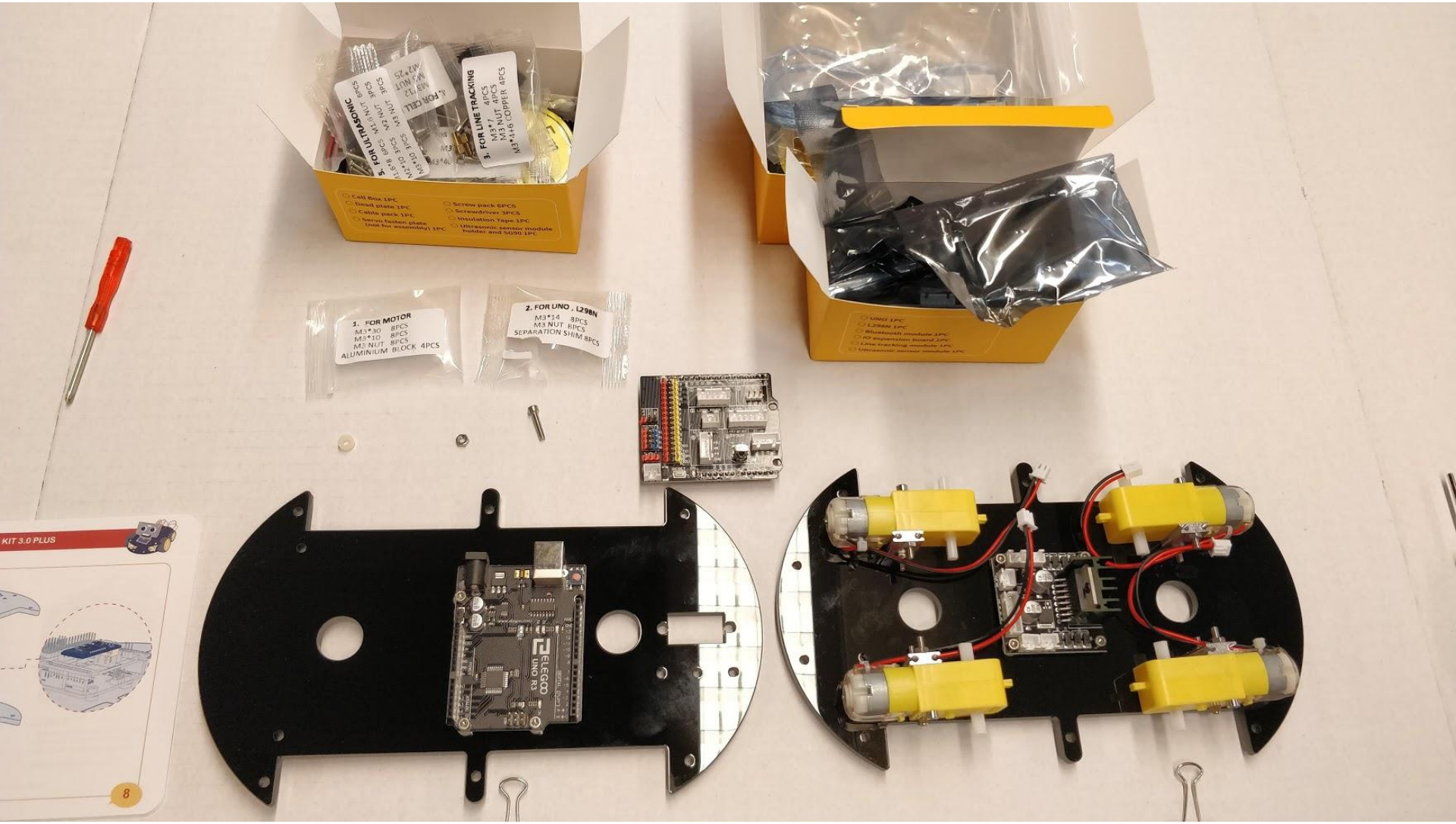


# Building the RC Car

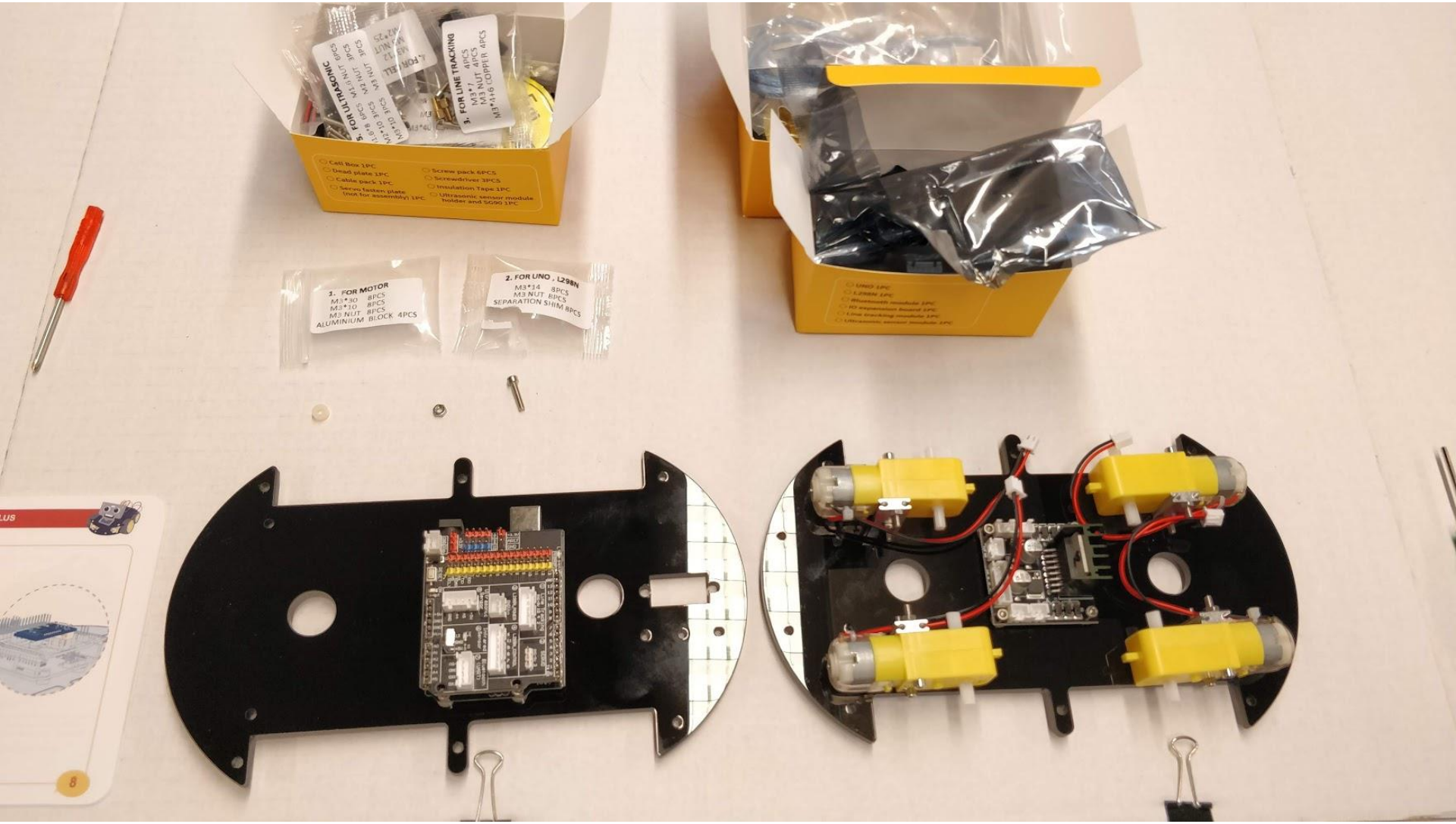
# Building the RC Car



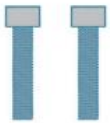
# Building the RC Car



# Building the RC Car

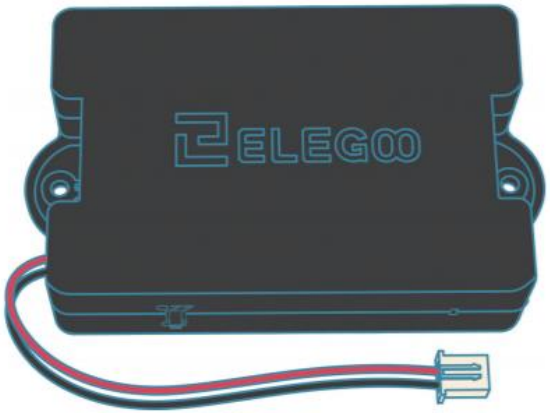


## SMART ROBOT CAR KIT 3.0 PLUS



① 2×M3*12 hexagon socket screws

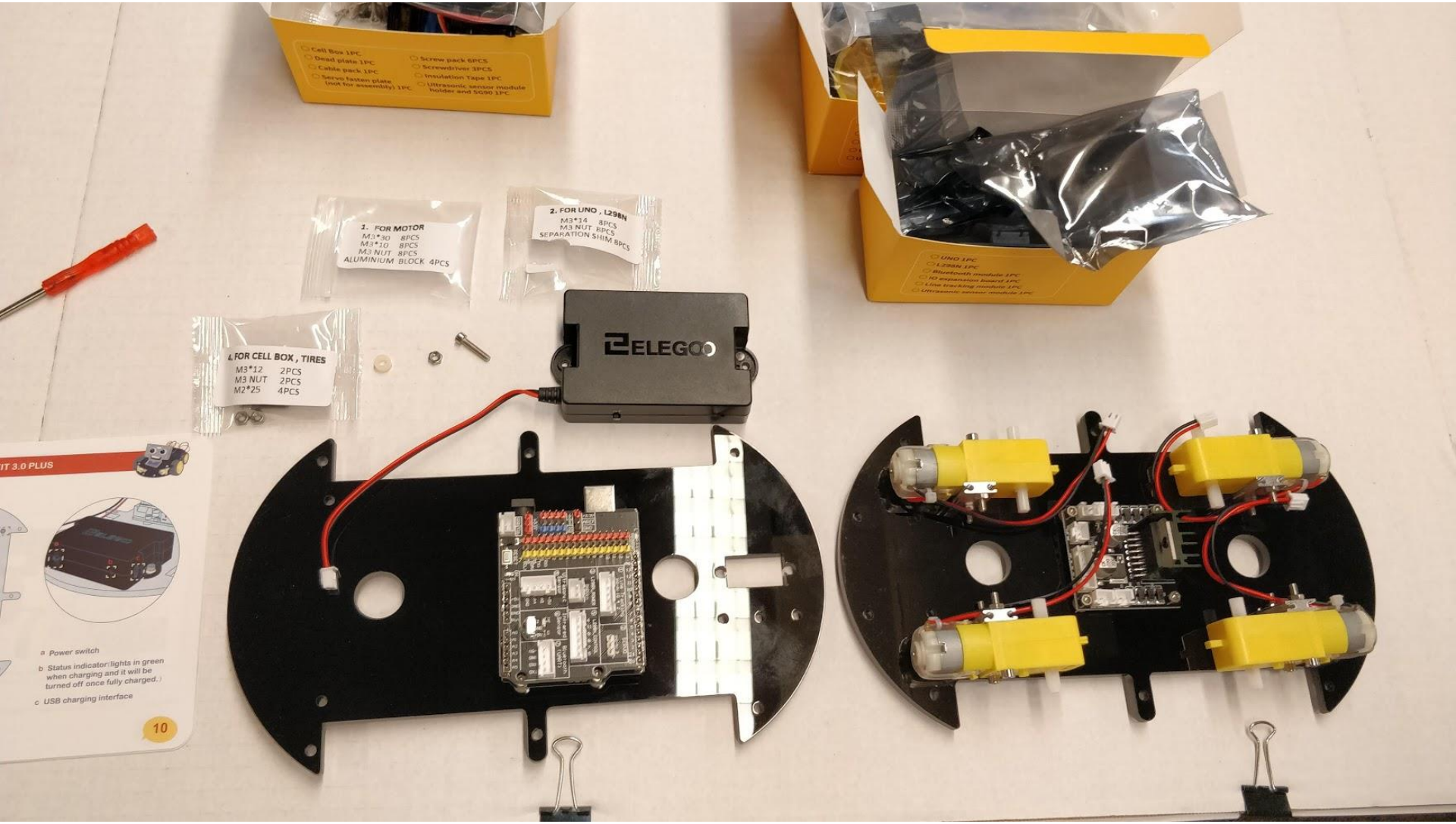
② 2×M3 nuts



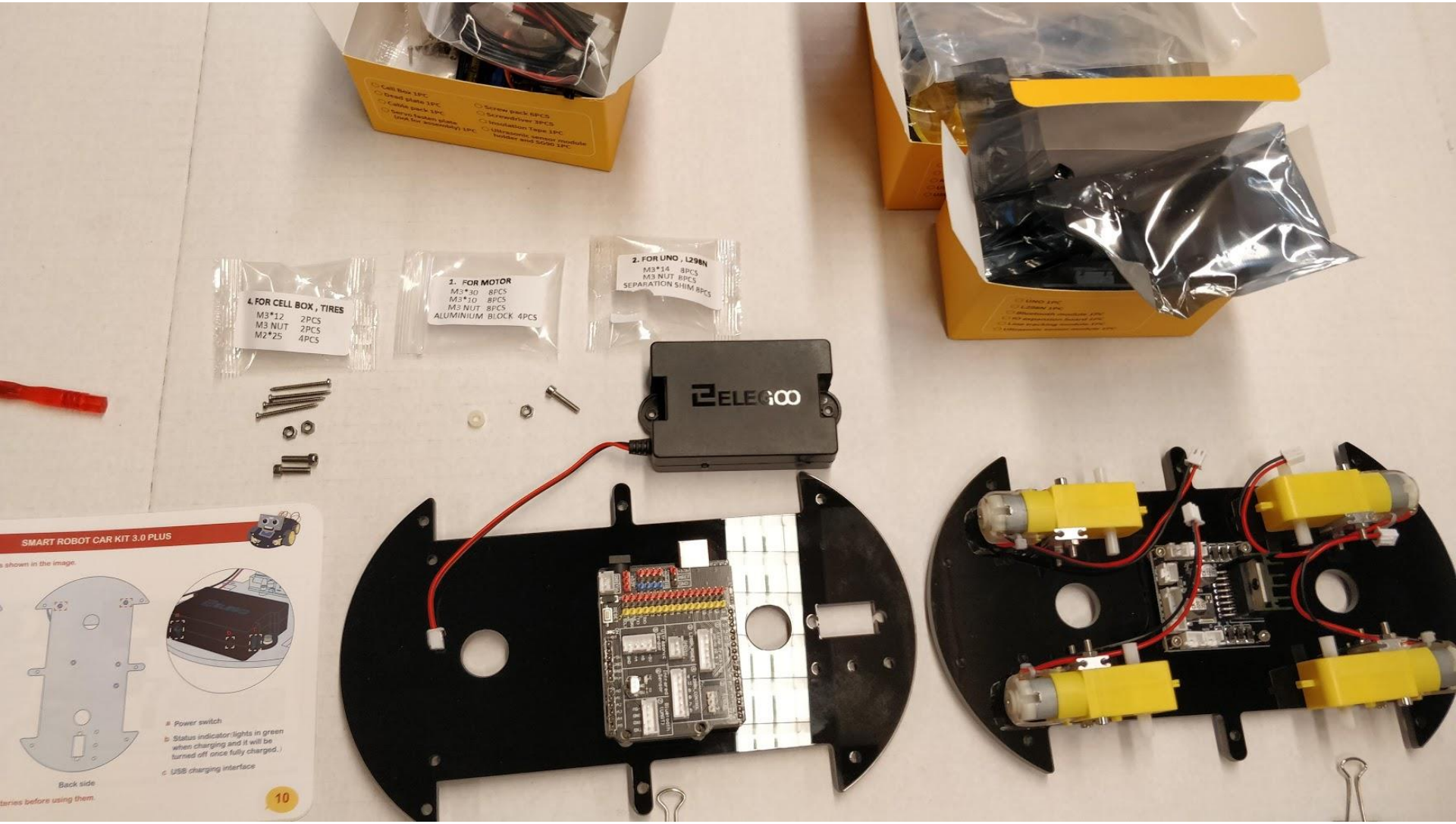

③ 1×Cell Box(Lithium Battery inside)

Take out ① ② from the bag with label "FOR CELL BOX、TIRES".

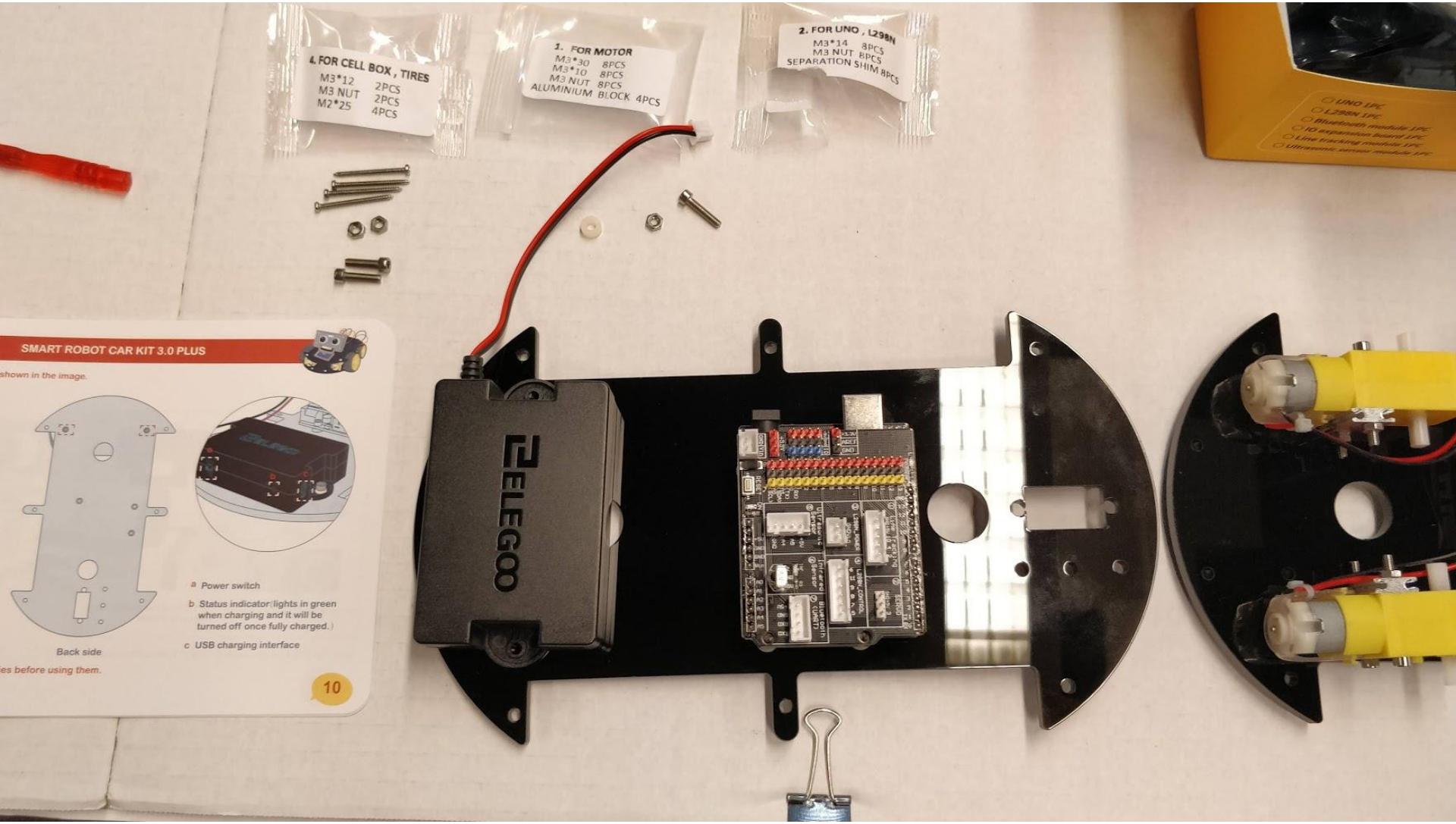
# Building the RC Car



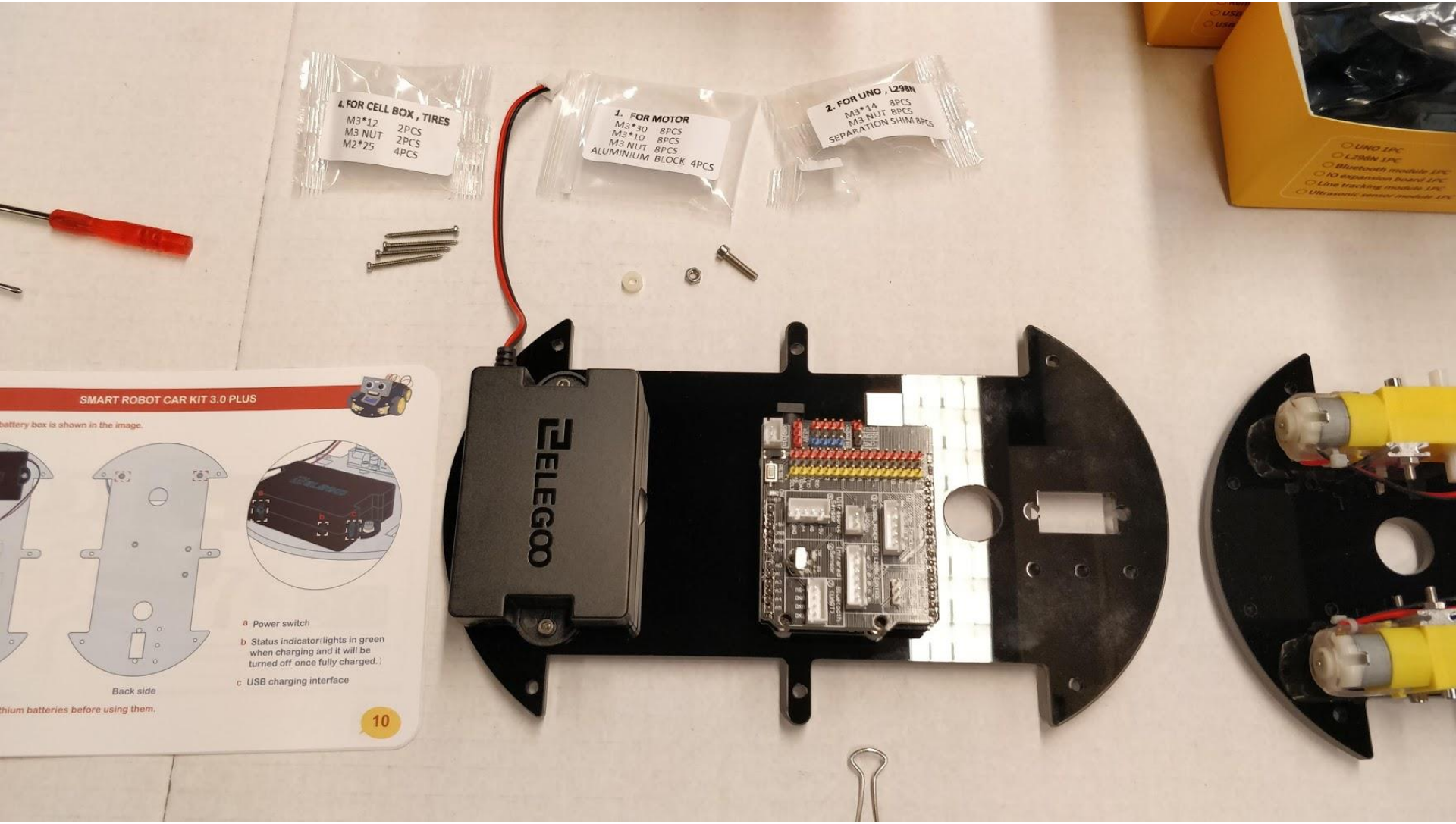
# Building the RC Car



# Building the RC Car

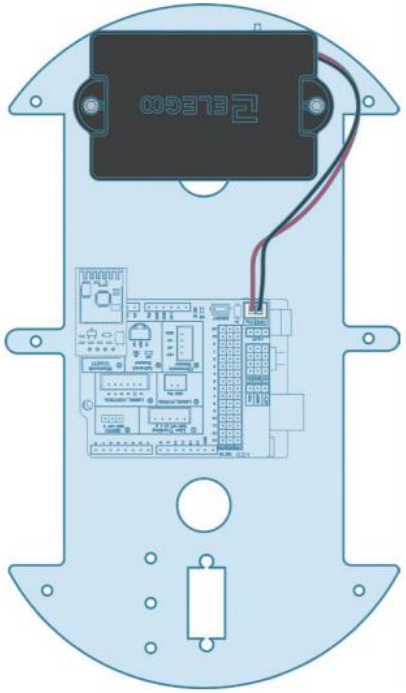


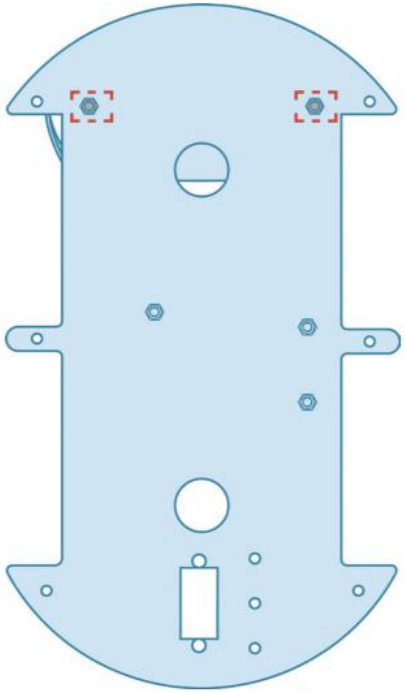
# Building the RC Car



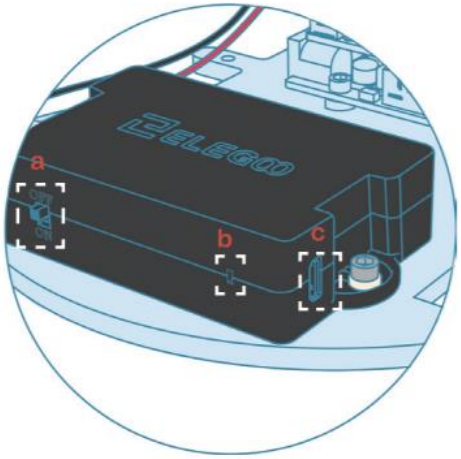
## SMART ROBOT CAR KIT 3.0 PLUS

The wiring of the battery box is shown in the image.





Back side



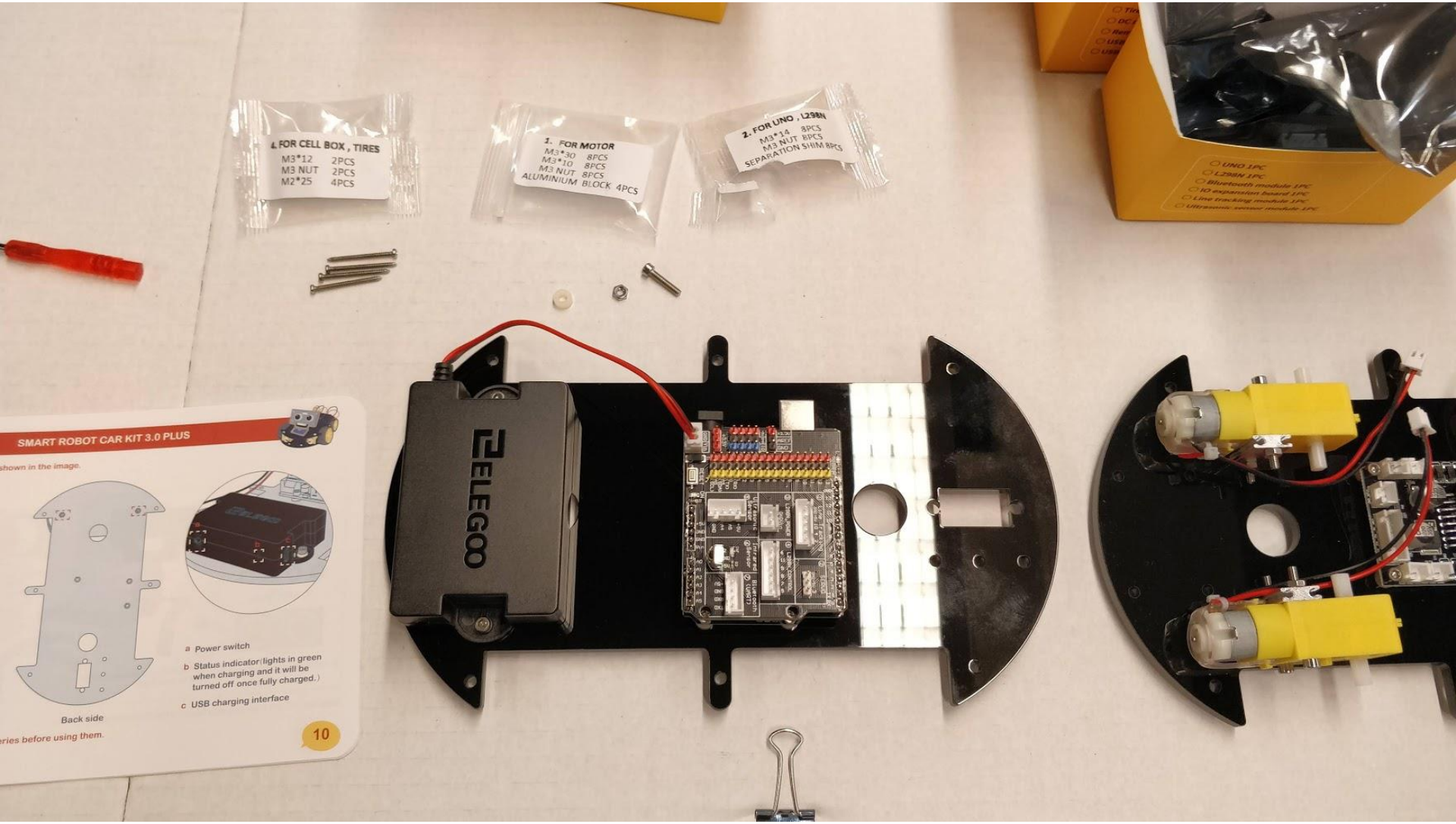

a Power switch

b Status indicator(lights in green when charging and it will be turned off once fully charged.)

c USB charging interface

Please fully charge the lithium batteries before using them.

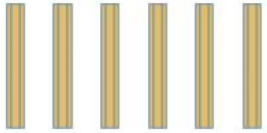
# Building the RC Car



# Building the RC Car



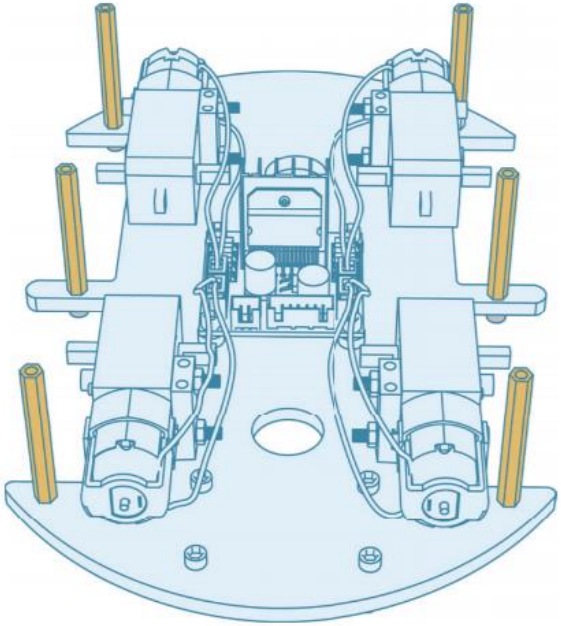
## SMART ROBOT CAR KIT 3.0 PLUS

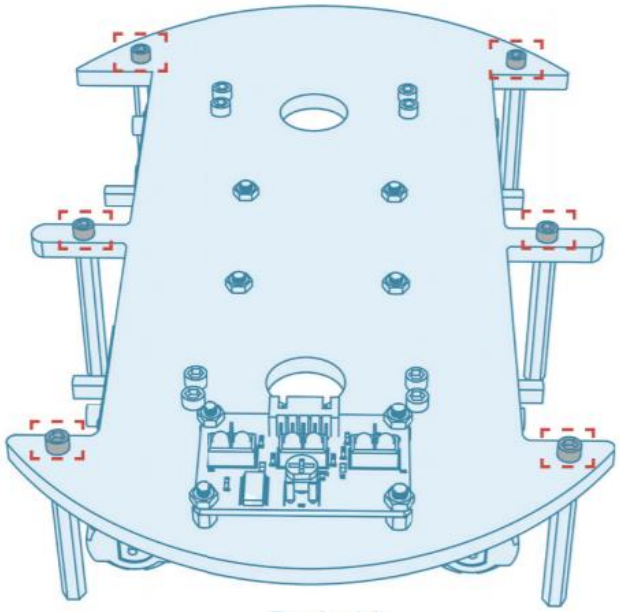


① 6×M3*40 double-pass copper cylinder

② 6×M3*10 hexagon socket screws

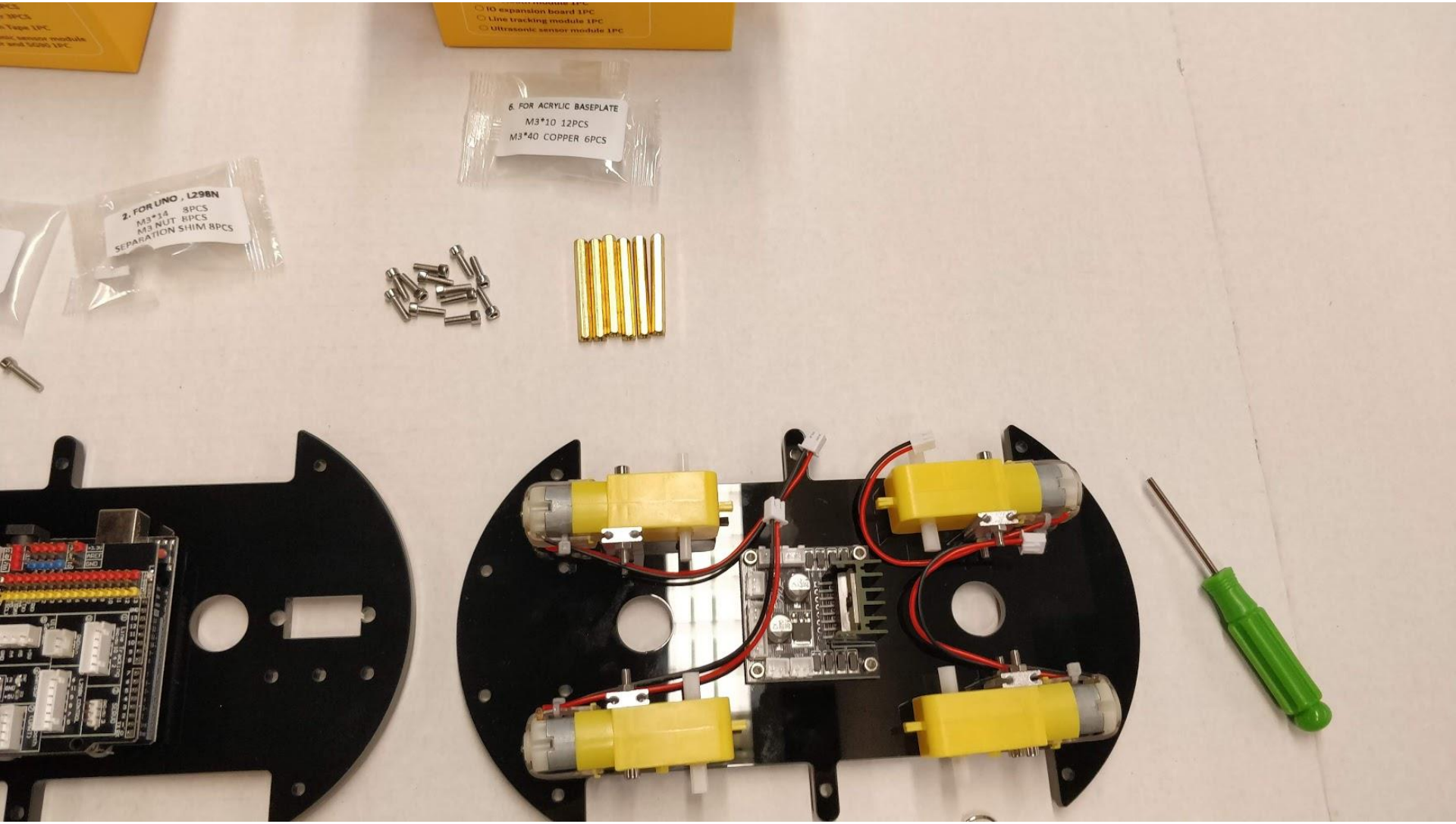
Take out ① ② from the bag with label " FOR ACRYLIC BASEPLATE".





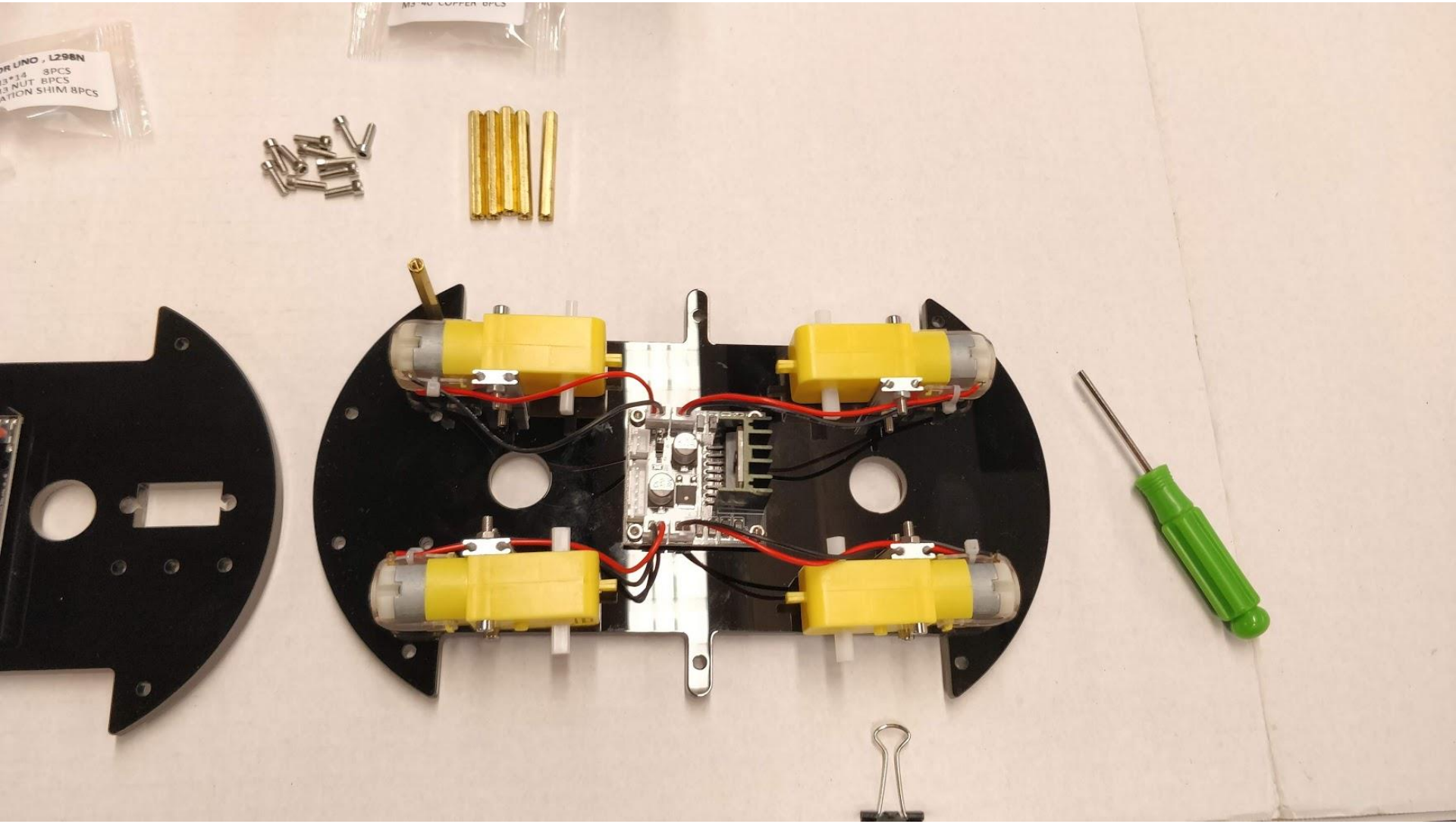
Back side

# Building the RC Car

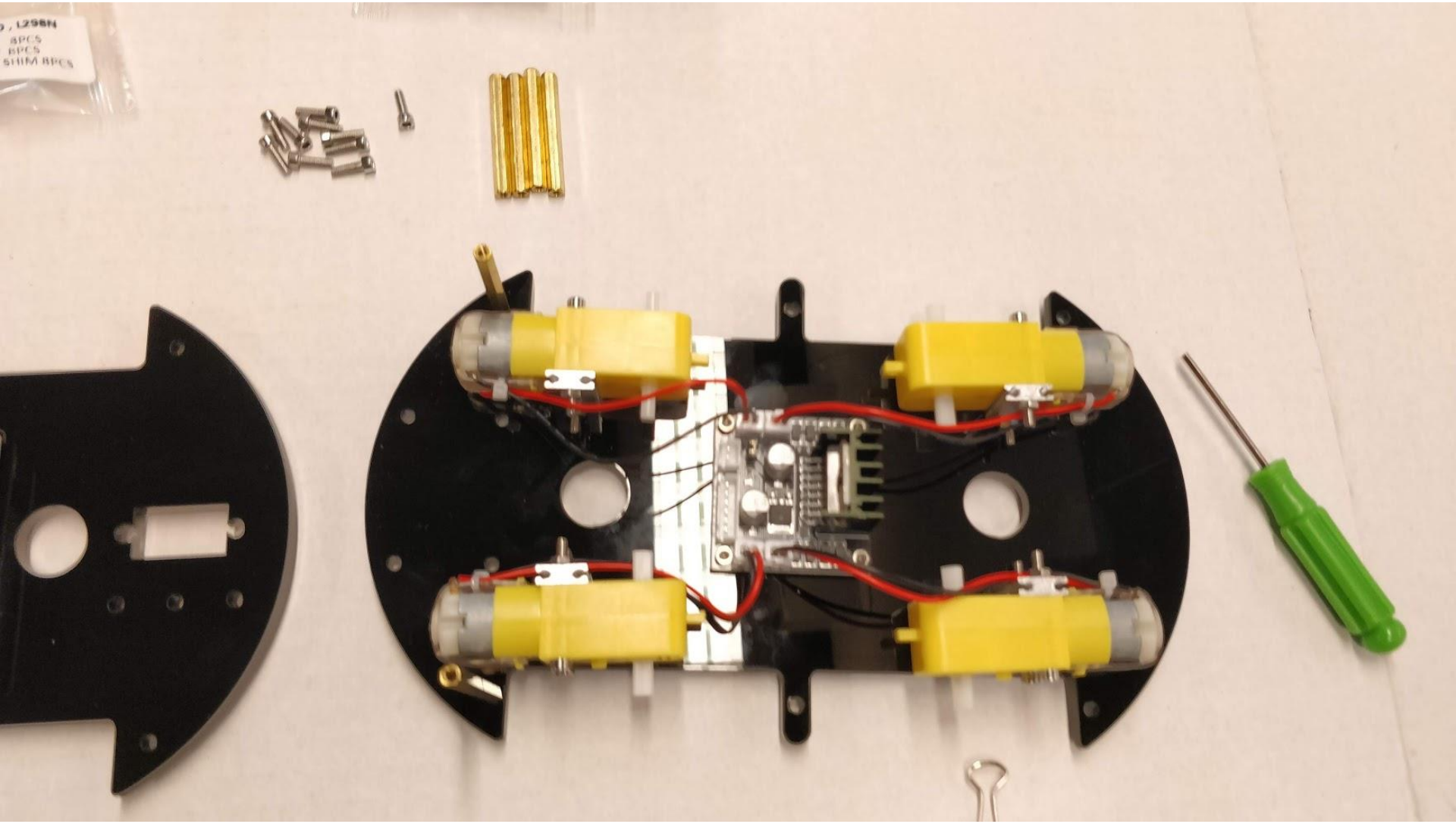


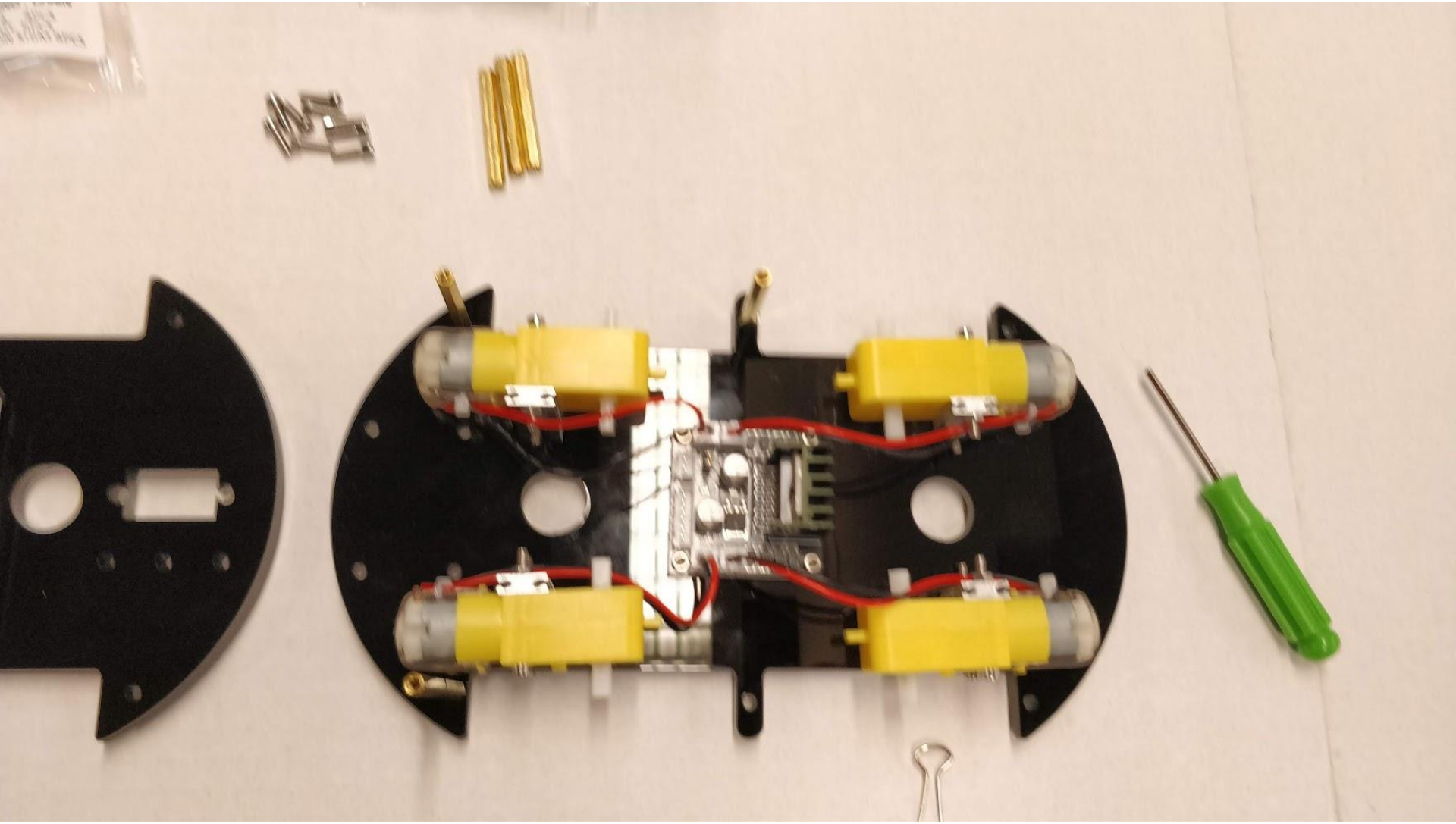
# Building the RC Car



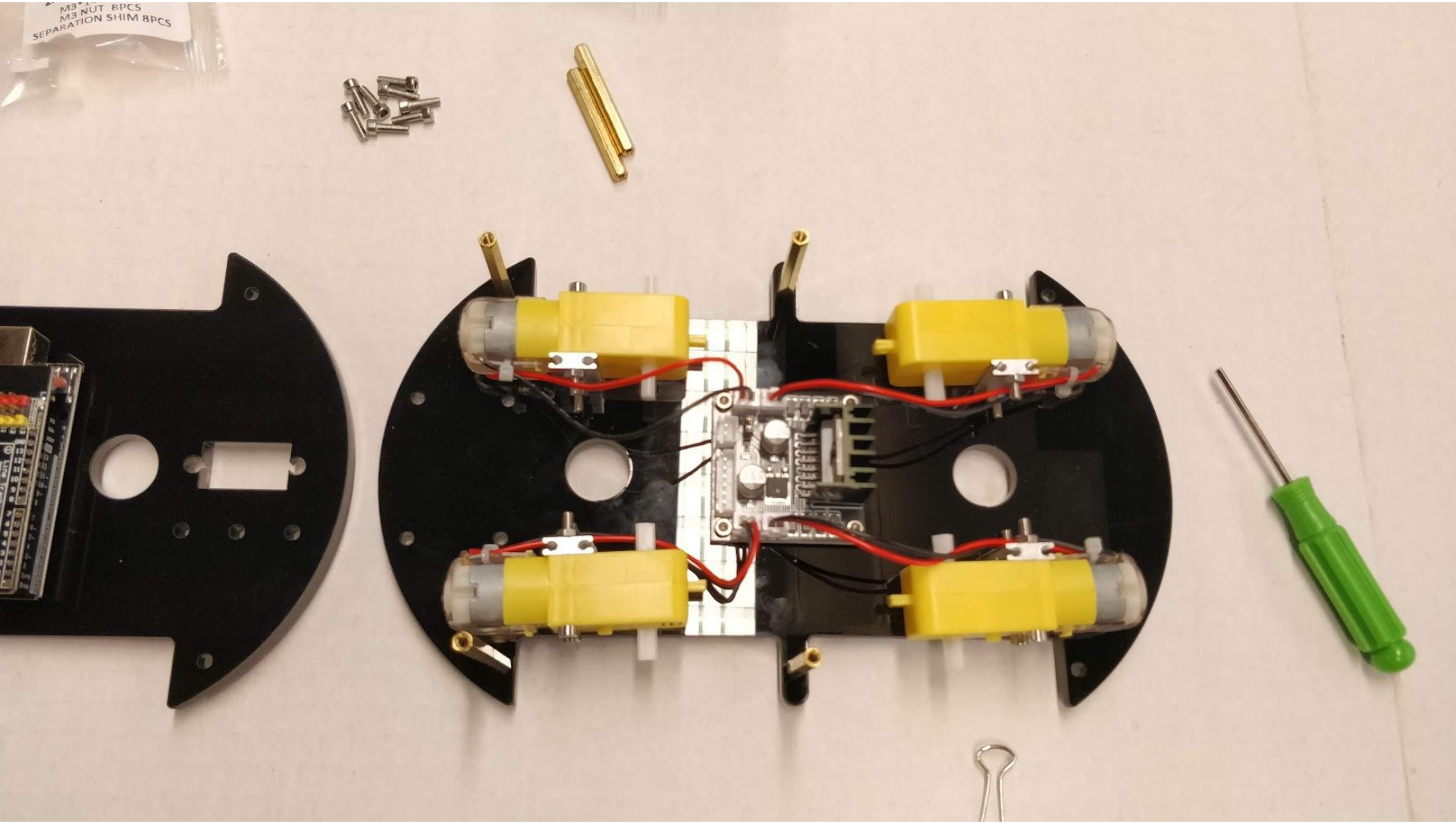


# Building the RC Car

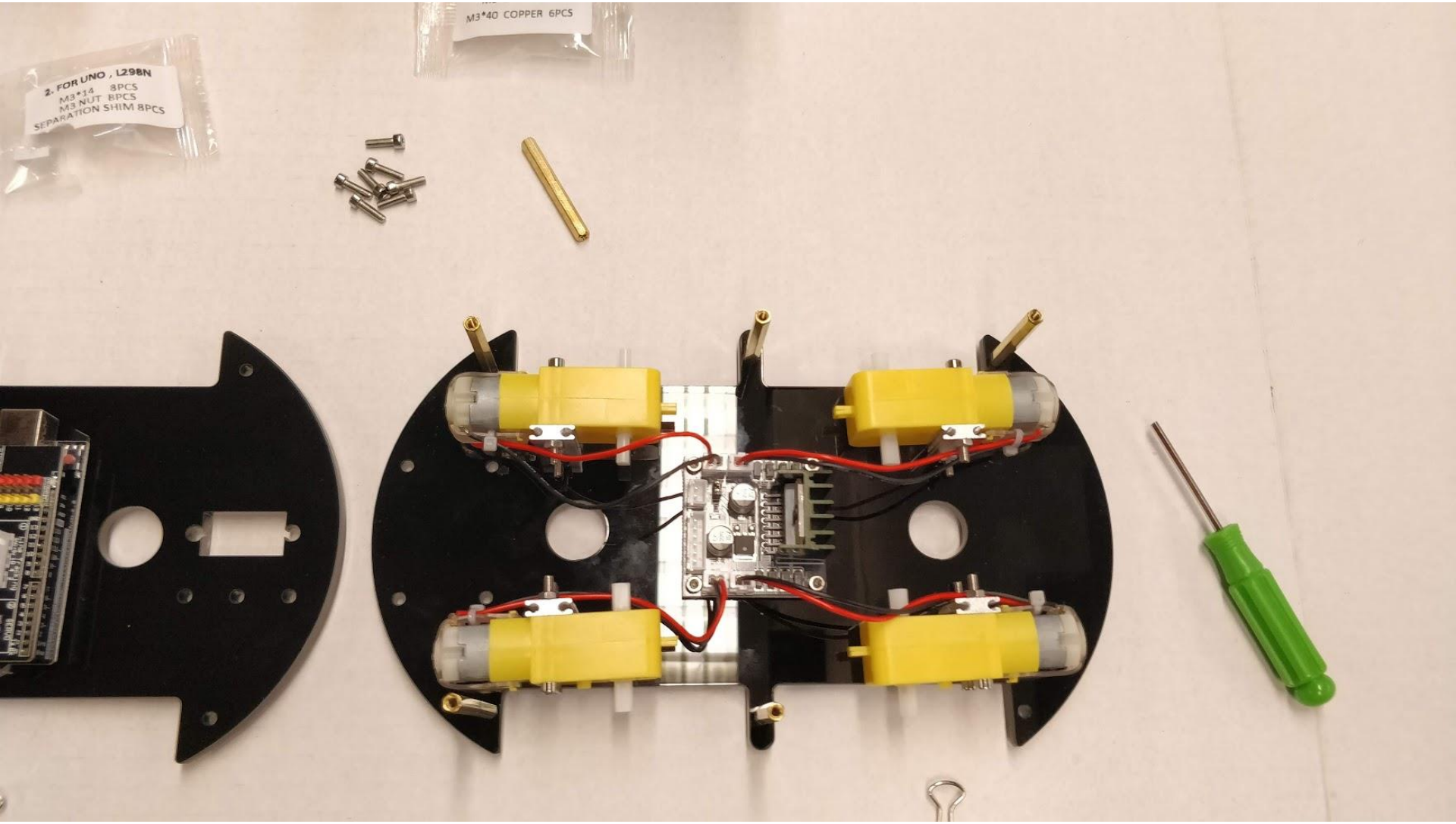




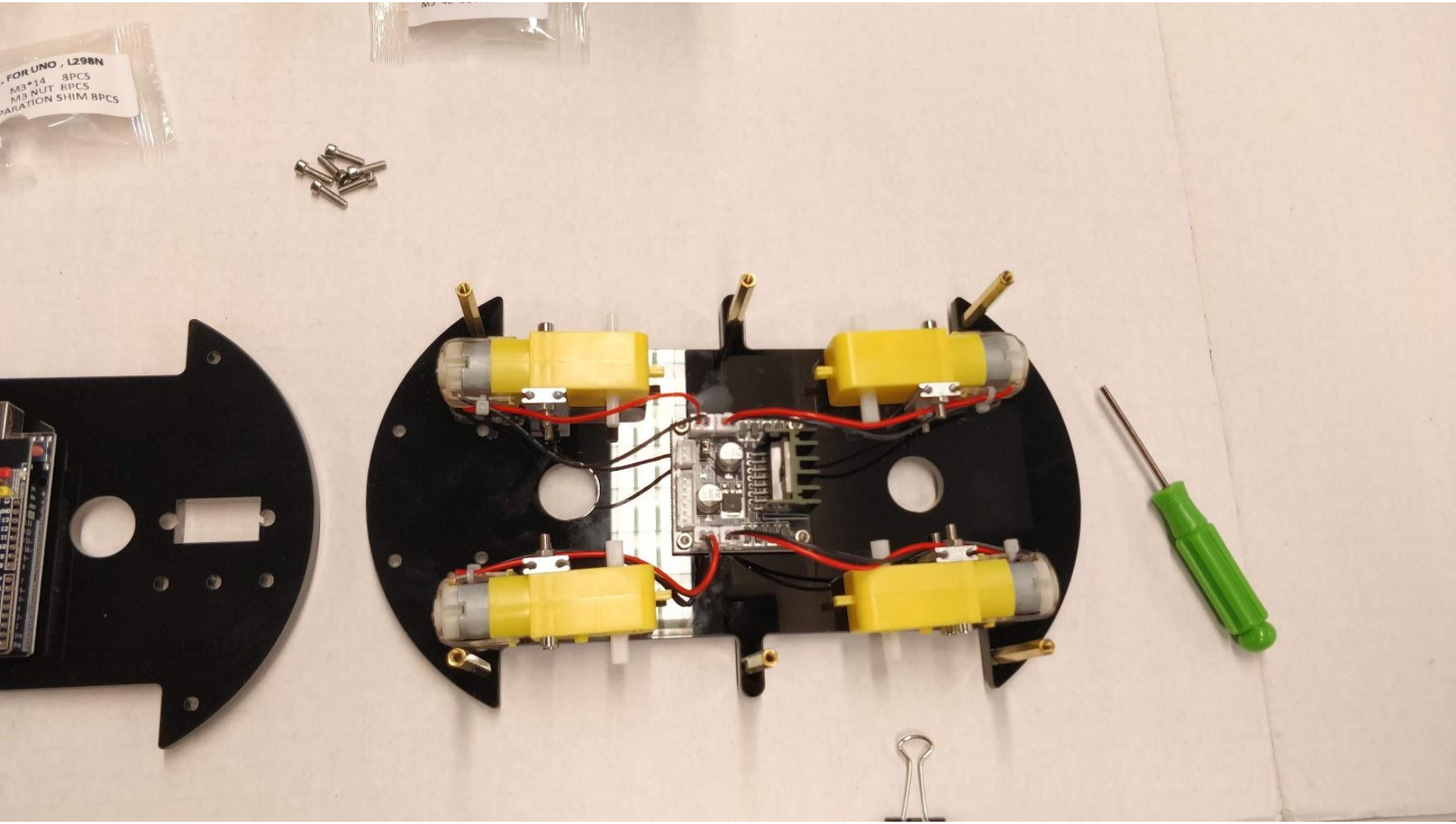
# Building the RC Car



# Building the RC Car

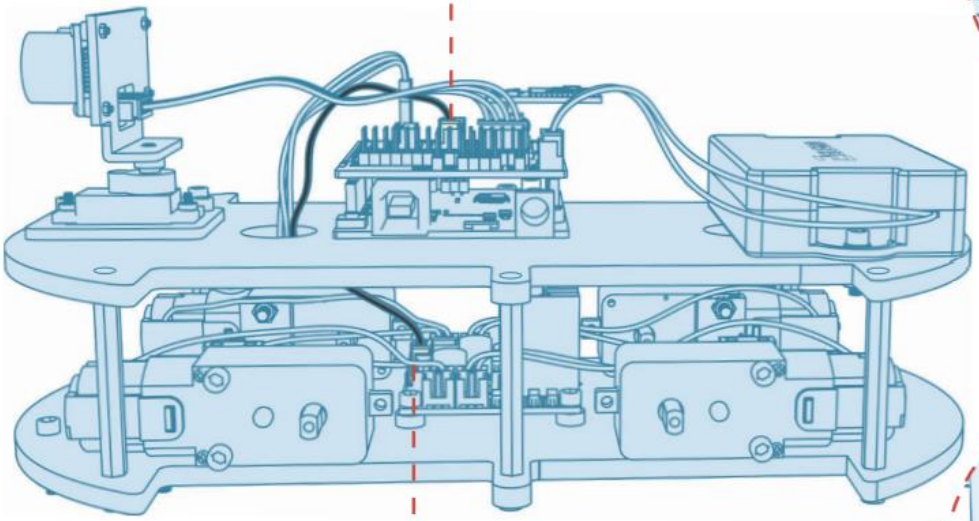


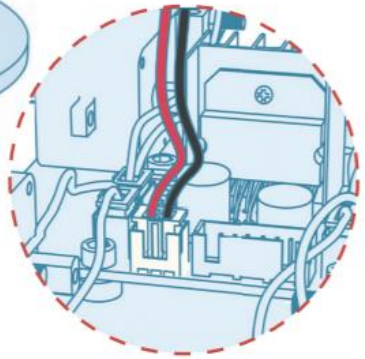
# Building the RC Car



## SMART ROBOT CAR KIT 3.0 PLUS

① 1 x2P cables

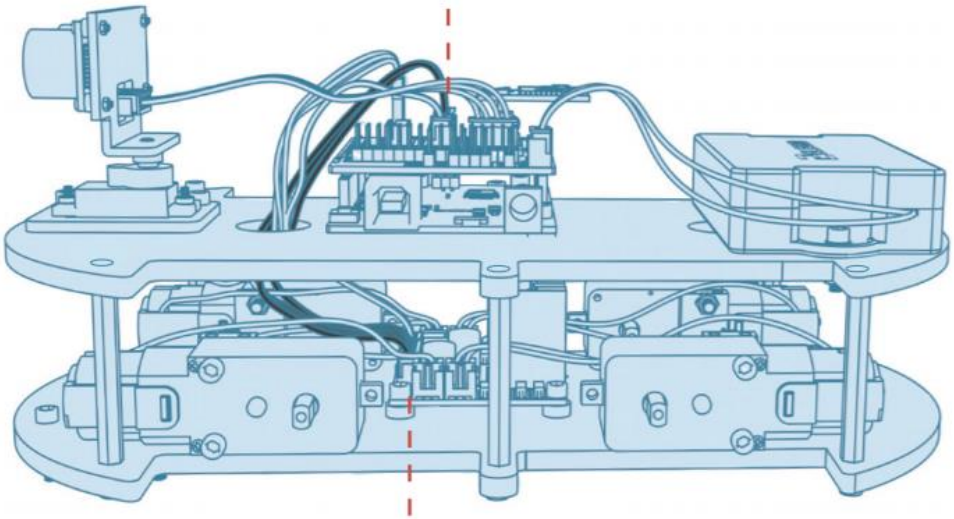


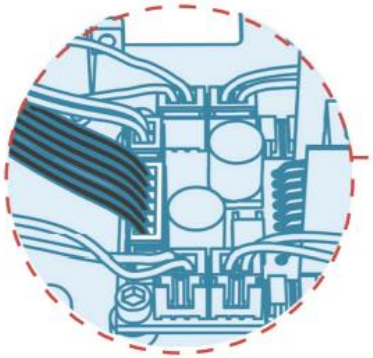


Connect the L298N to the expansion board.

## SMART ROBOT CAR KIT 3.0 PLUS

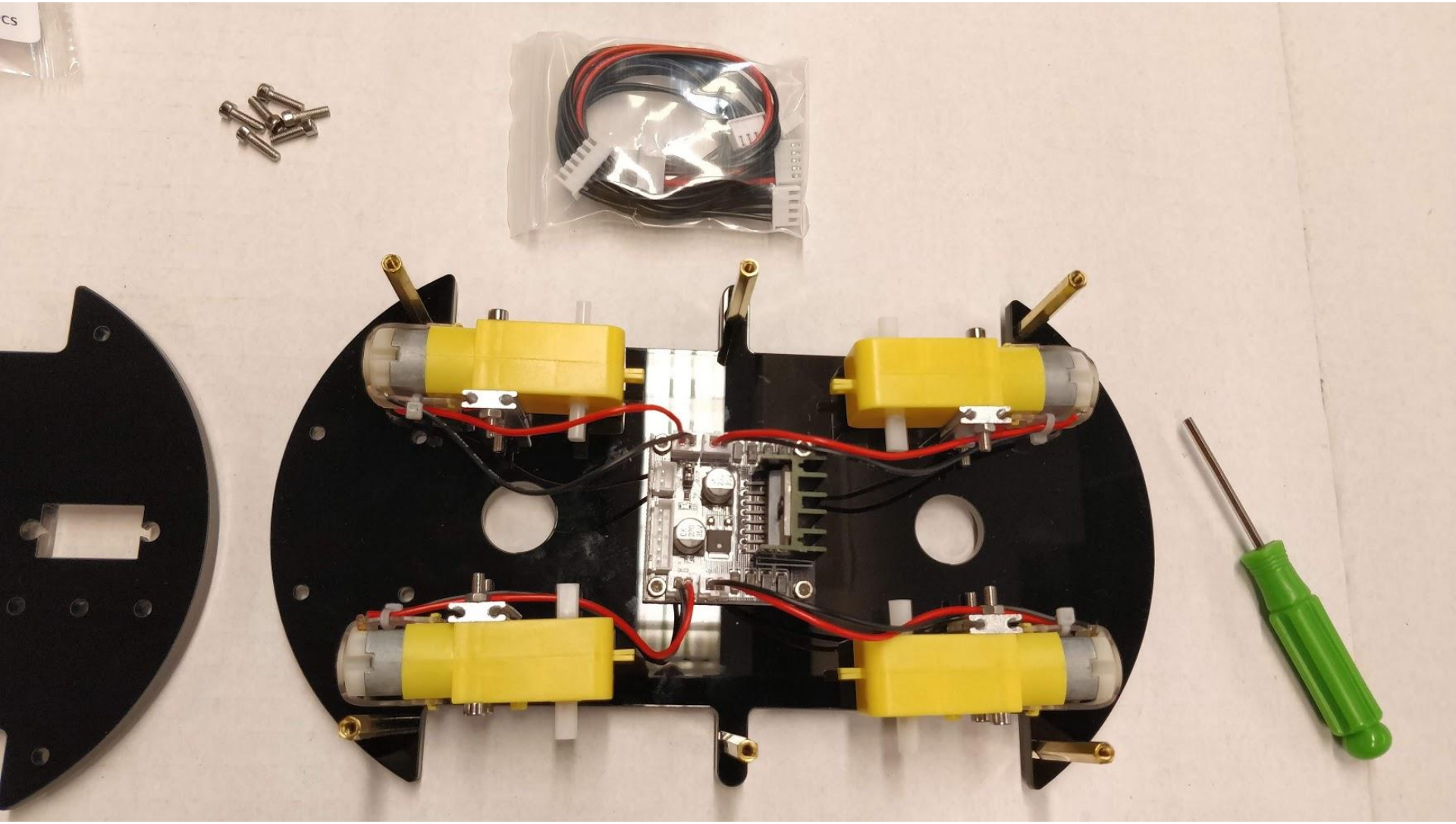
① 1 x6P cables



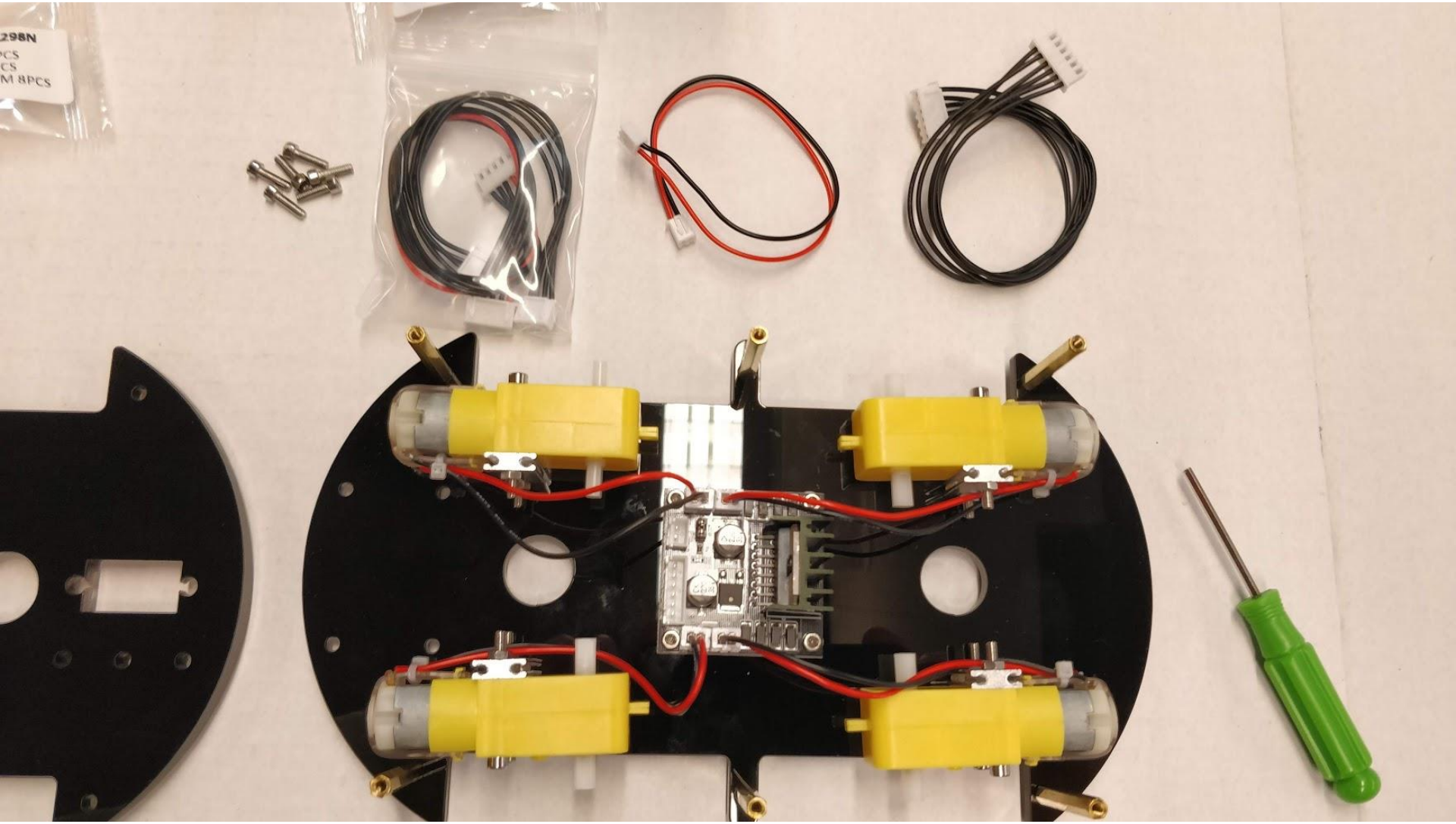


Connect the L298N to the expansion board.

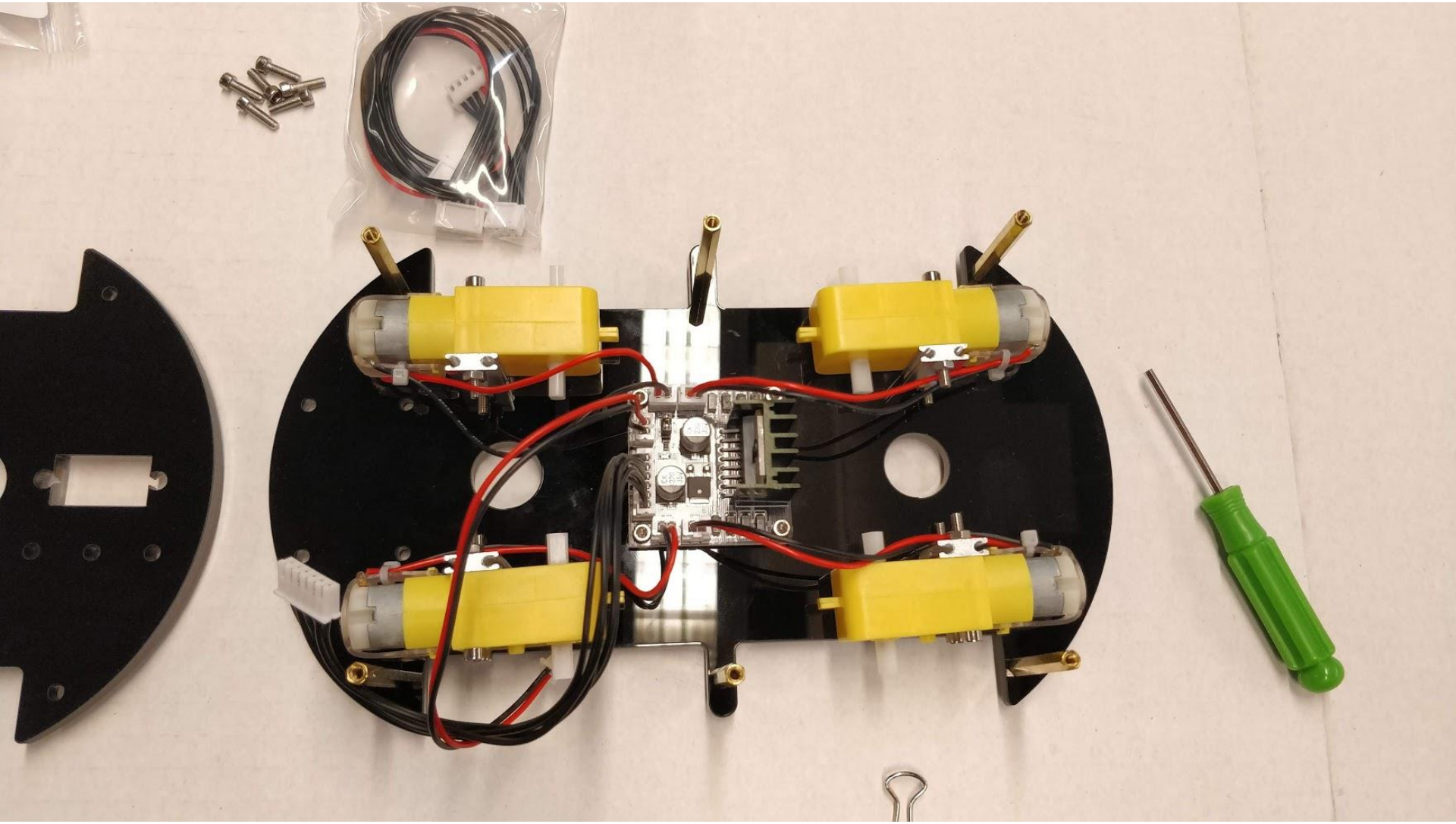
# Building the RC Car



# Building the RC Car



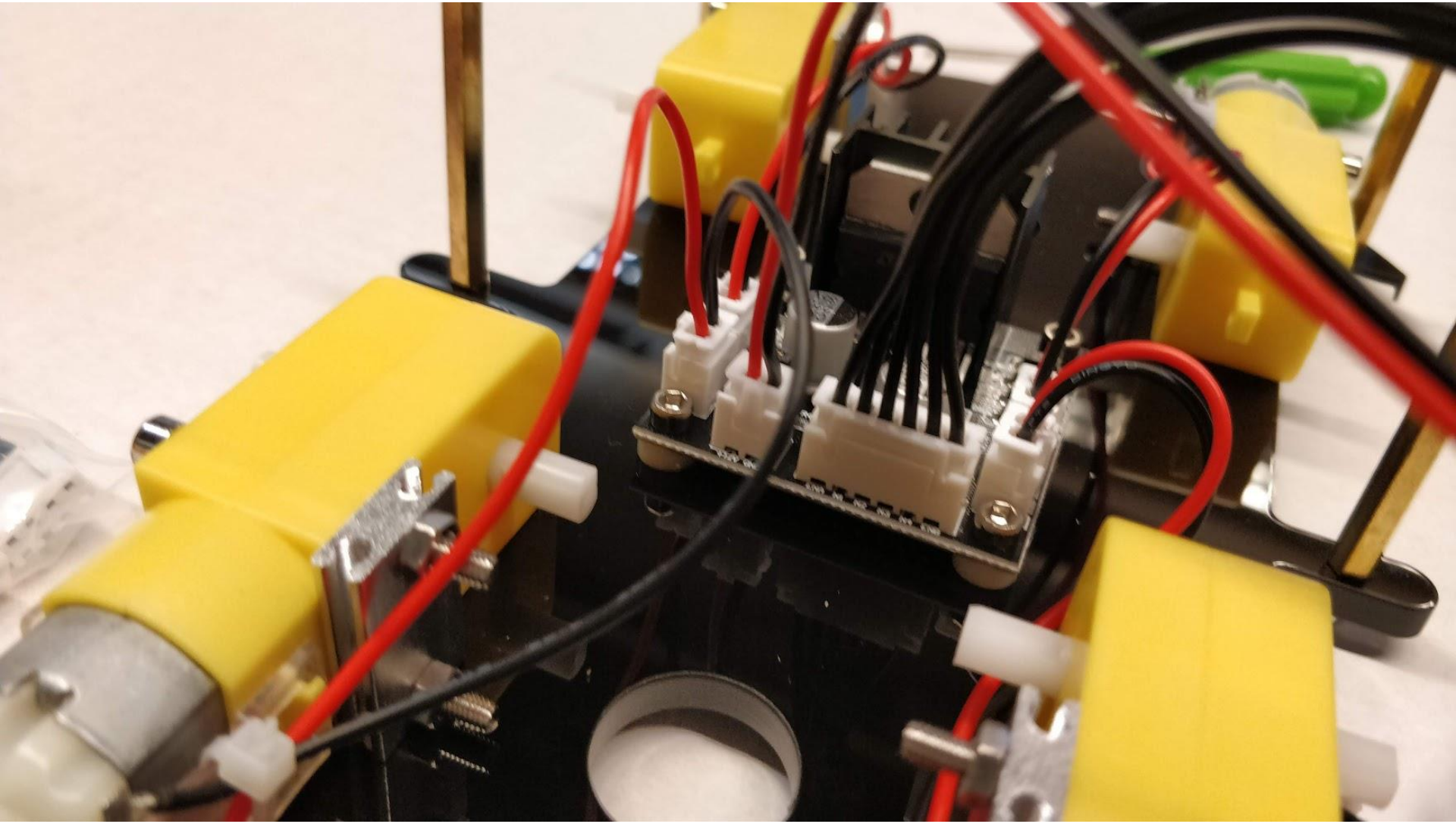
# Building the RC Car



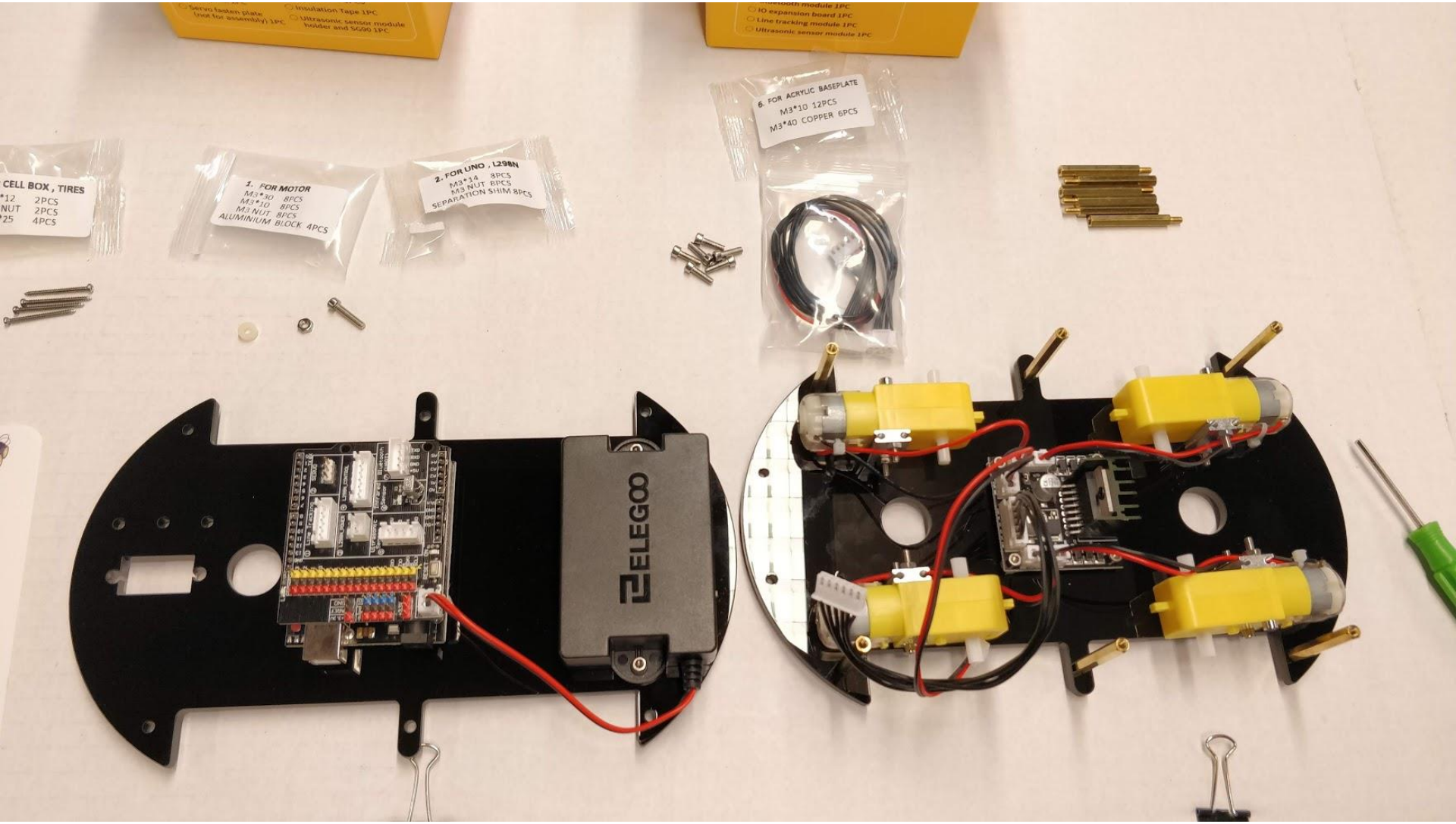
Joint Institute for Computational Sciences

# Building the RC Car

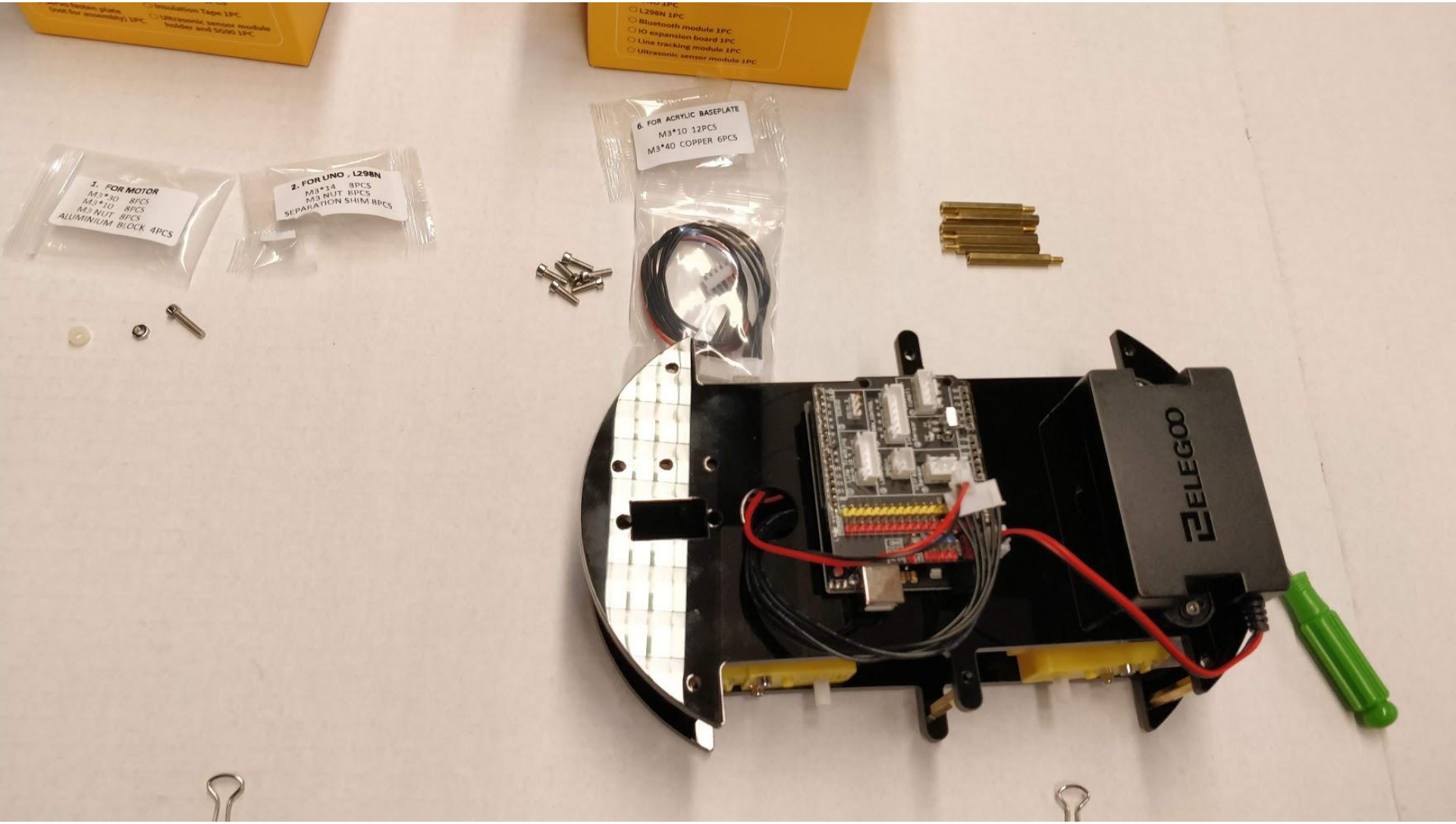
INNOVATIVE COMPUTING LABORATORY



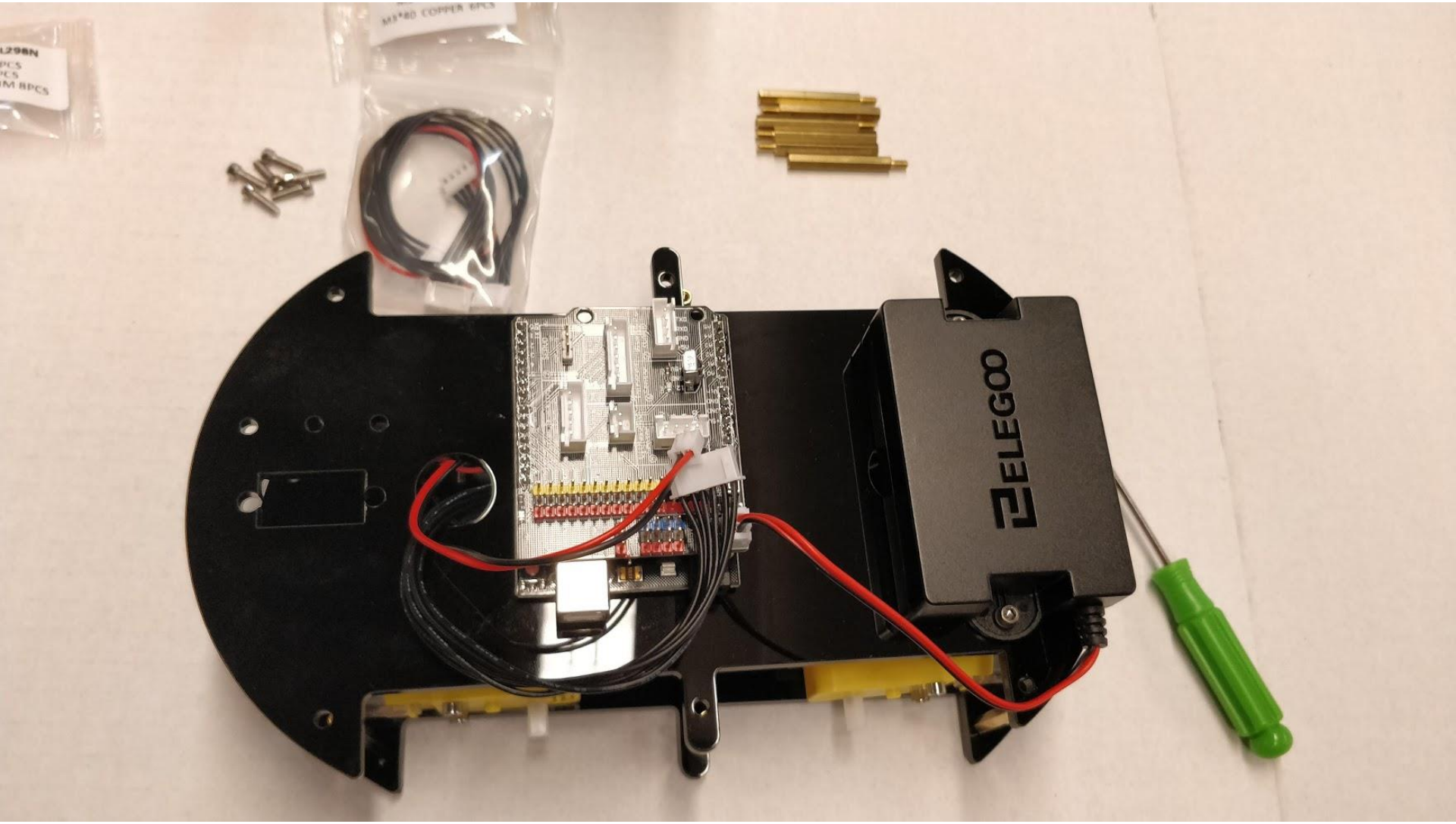
# Building the RC Car



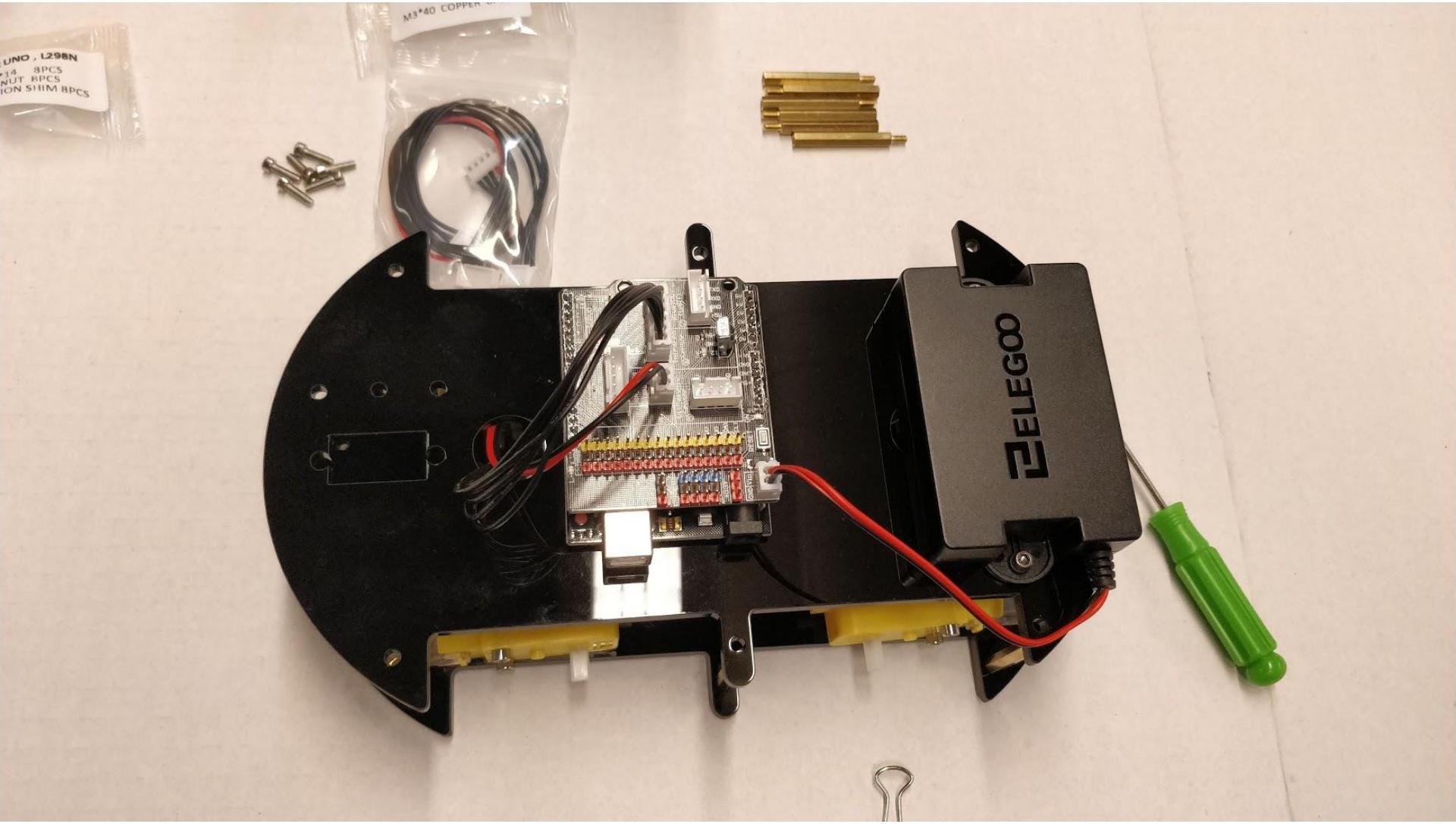
# Building the RC Car



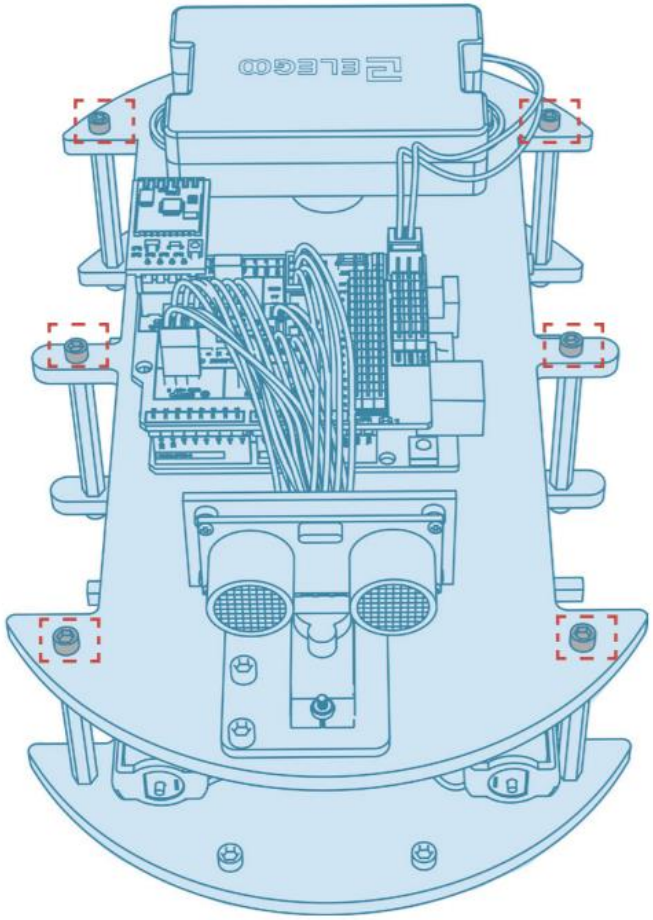
# Building the RC Car

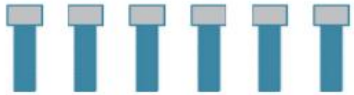


# Building the RC Car



## SMART ROBOT CAR KIT 3.0 PLUS

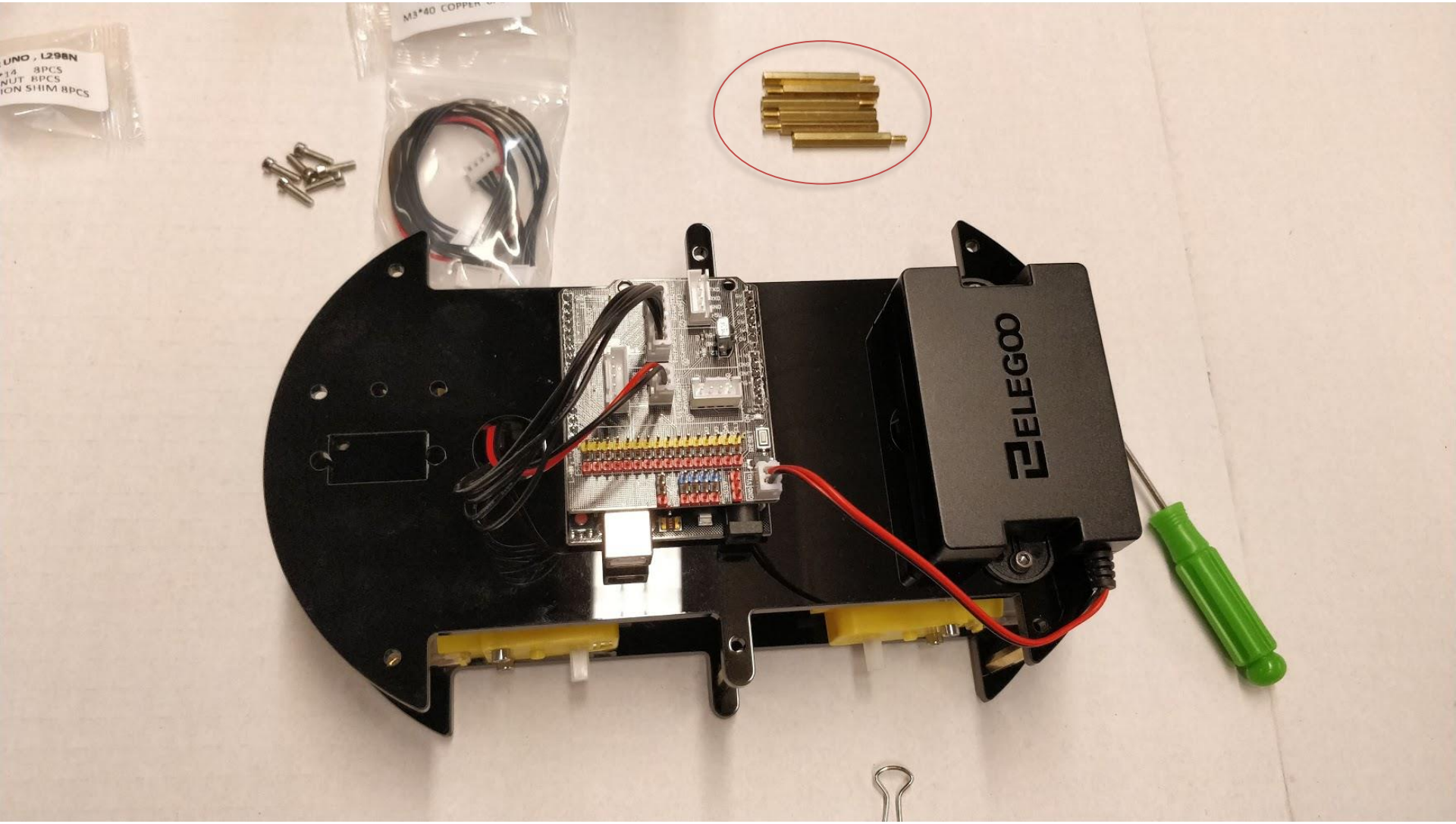




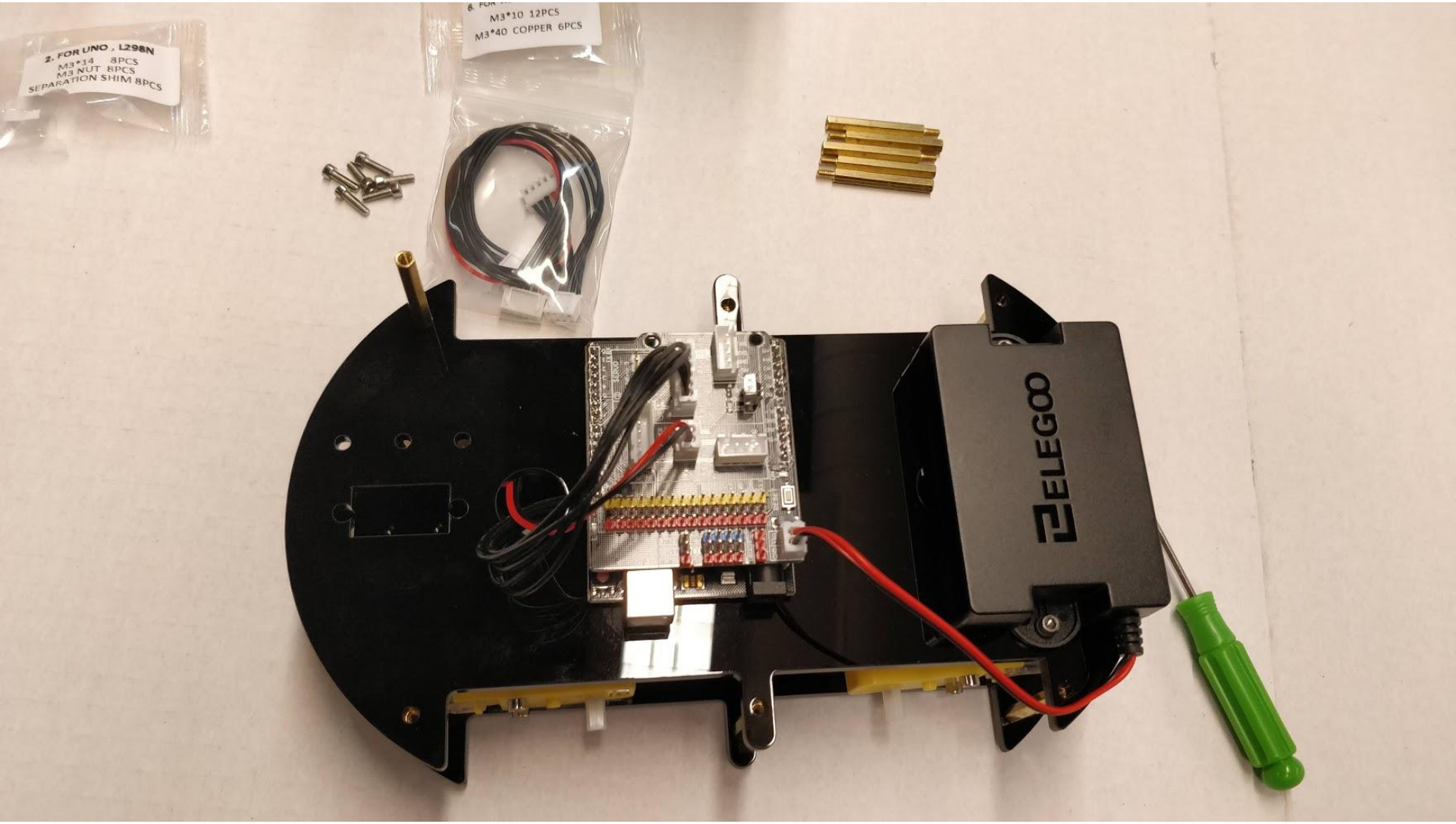
① 6×M3*10 hexagon socket screws

Take out ① from the bag with label " FOR ACRYLIC BASEPLATE".

# Building the RC Car

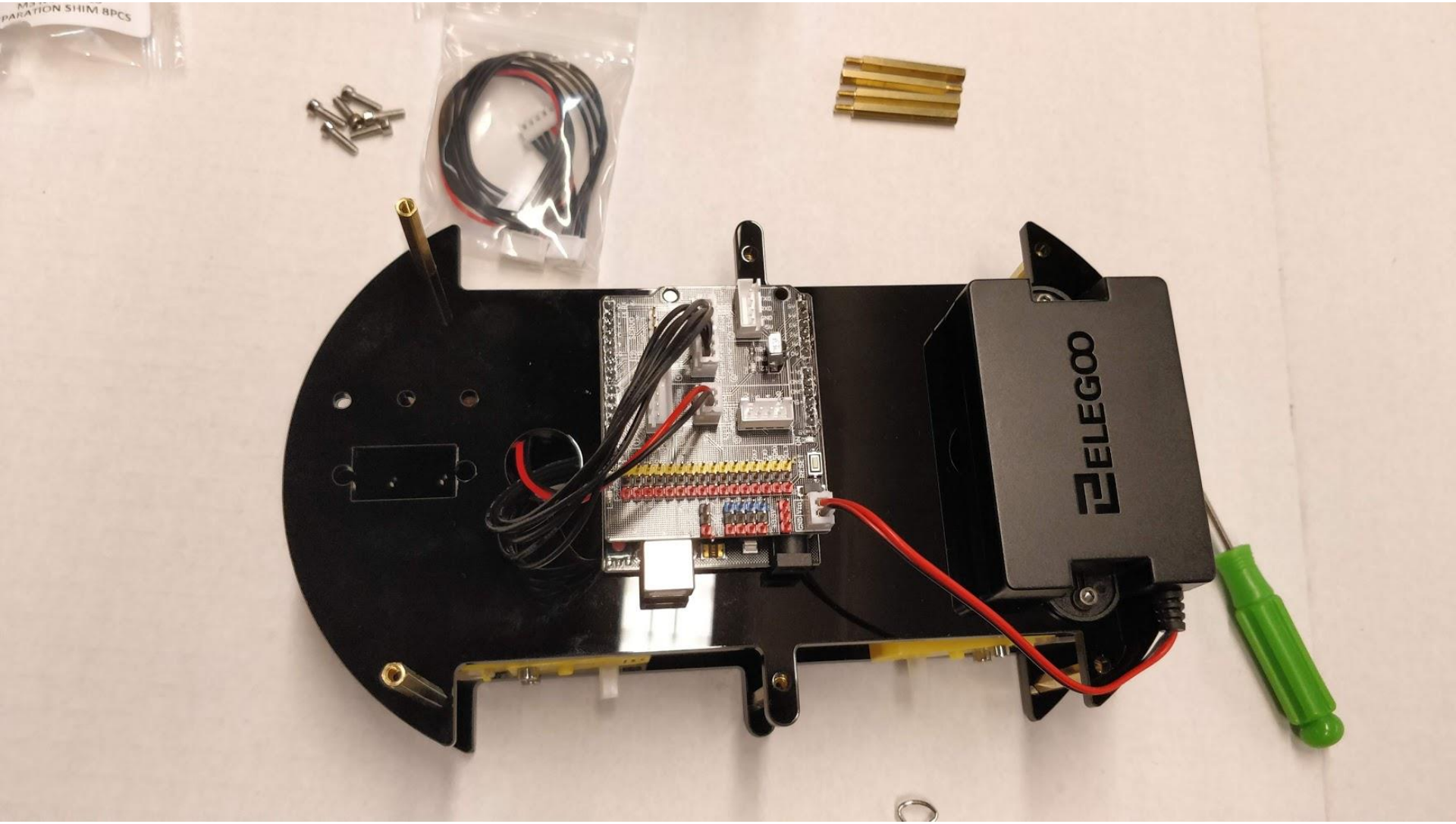


# Building the RC Car

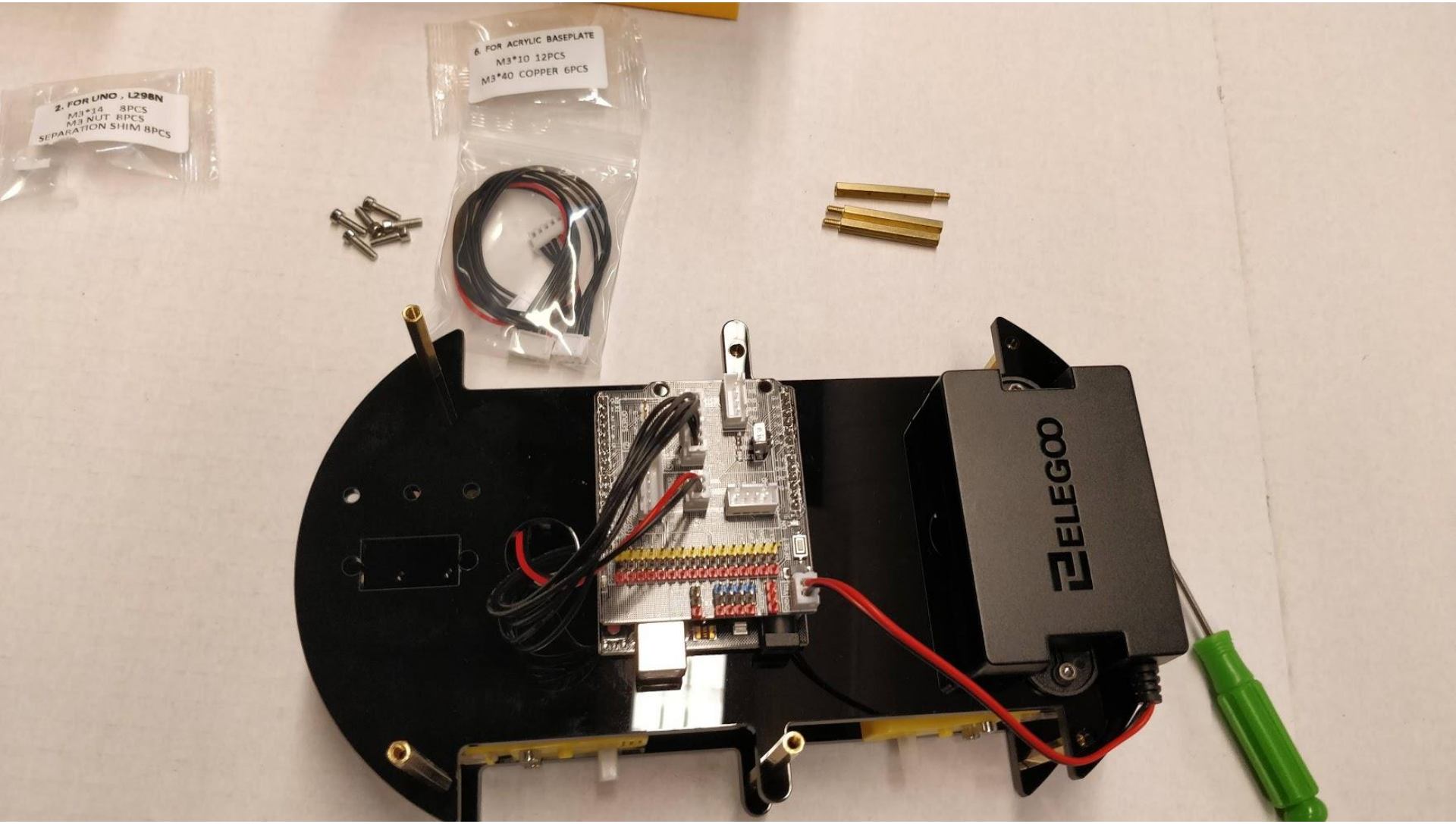


# Building the RC Car

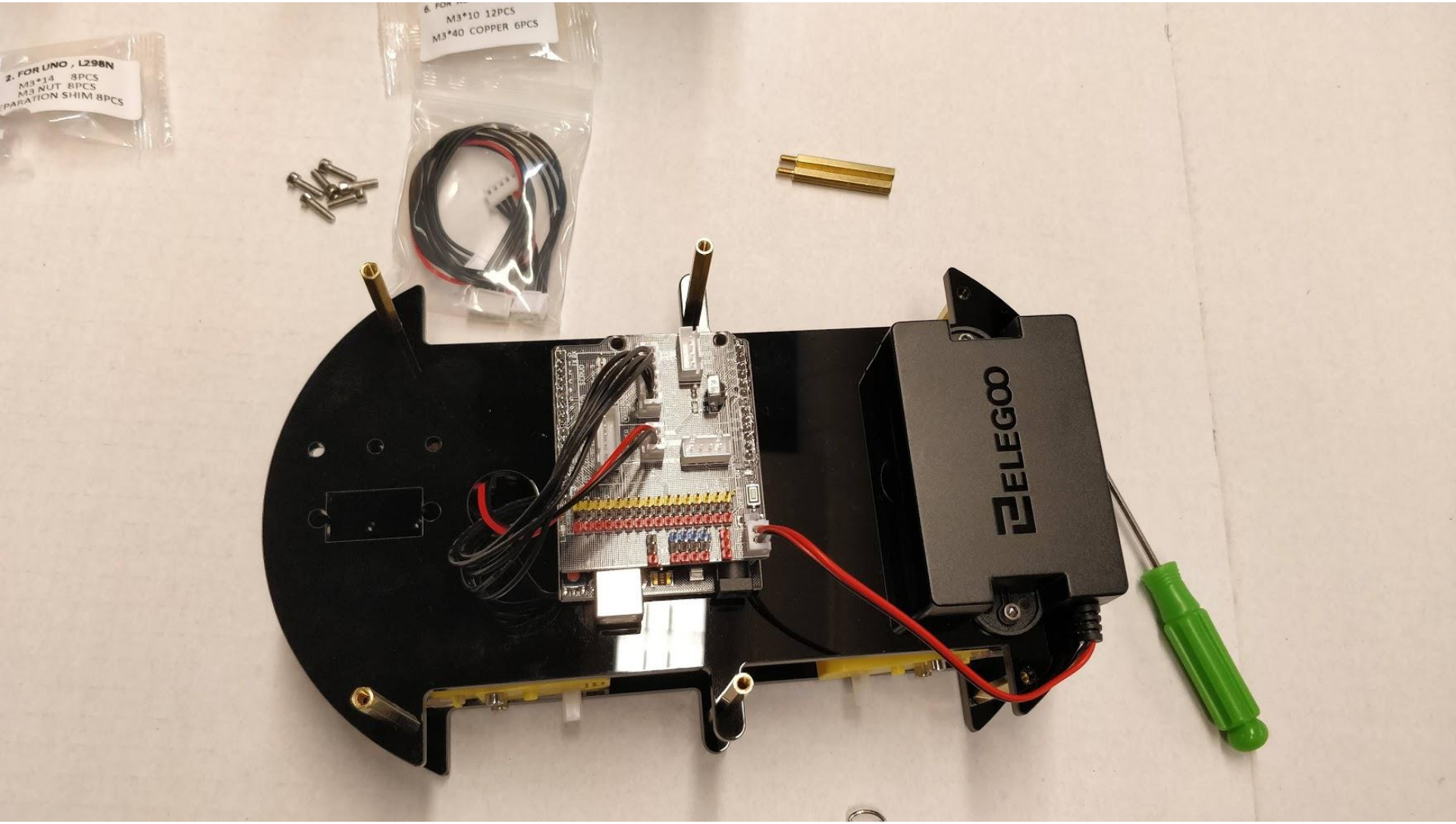




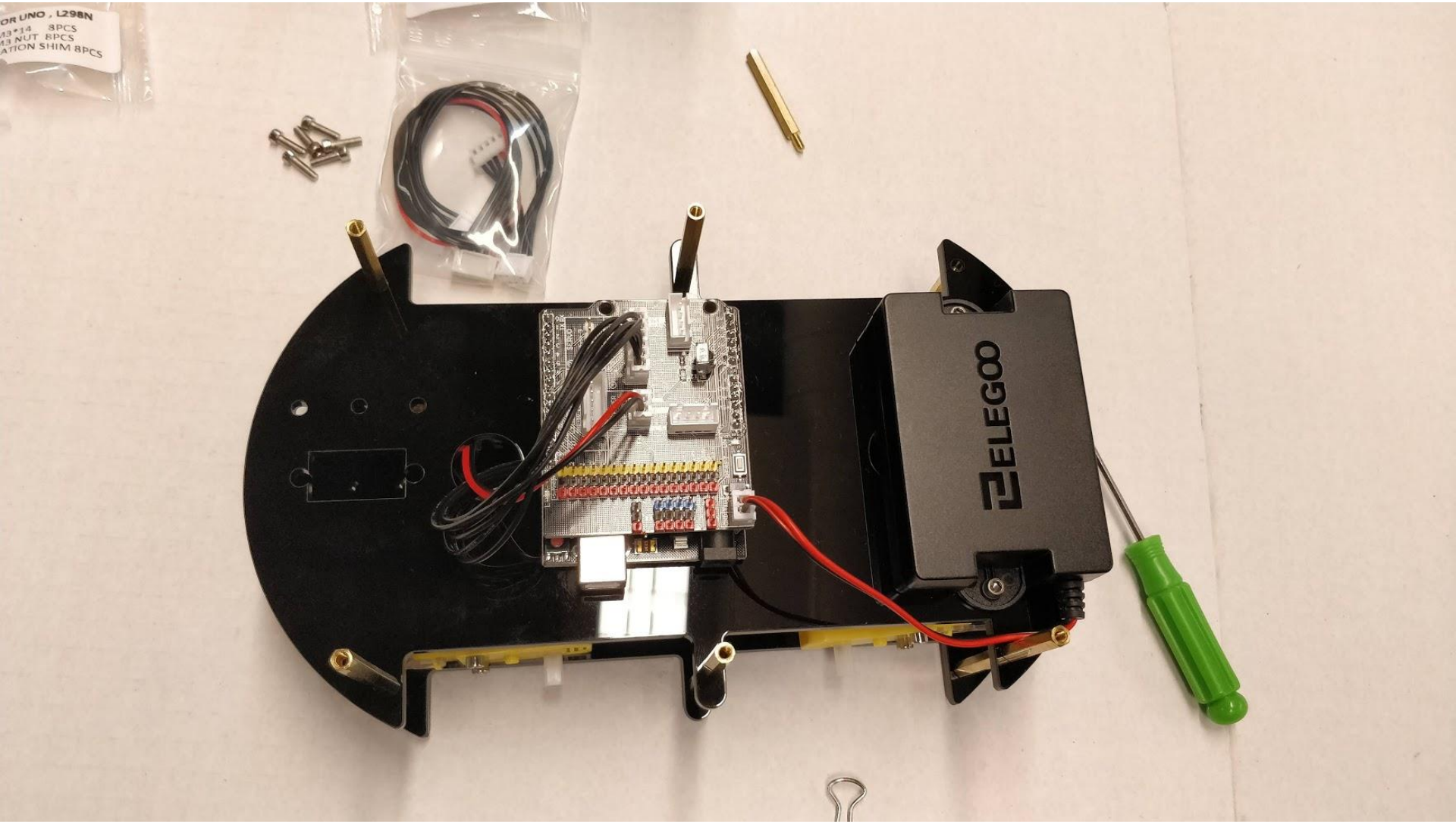
# Building the RC Car



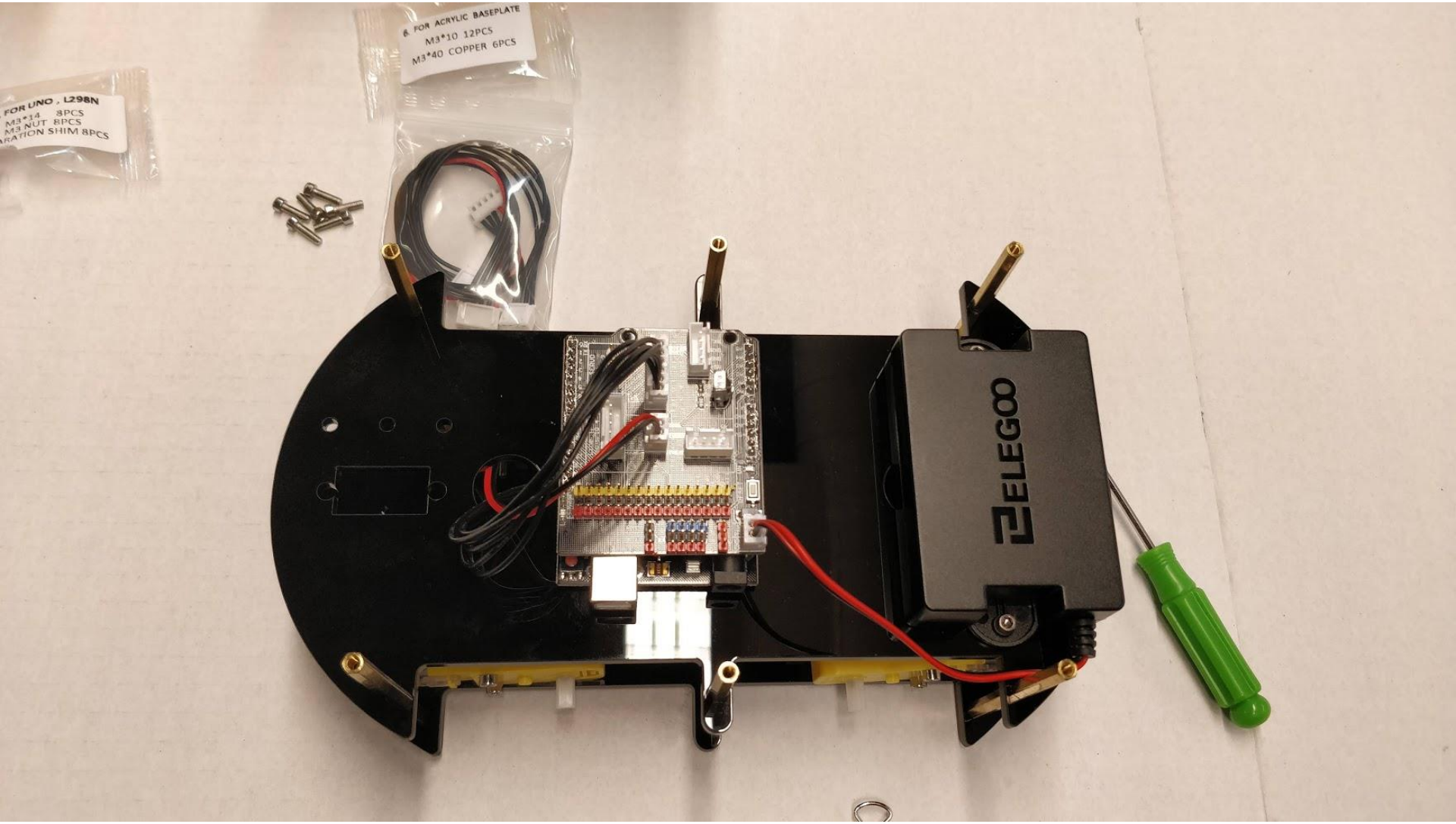
# Building the RC Car



# Building the RC Car



# Building the RC Car

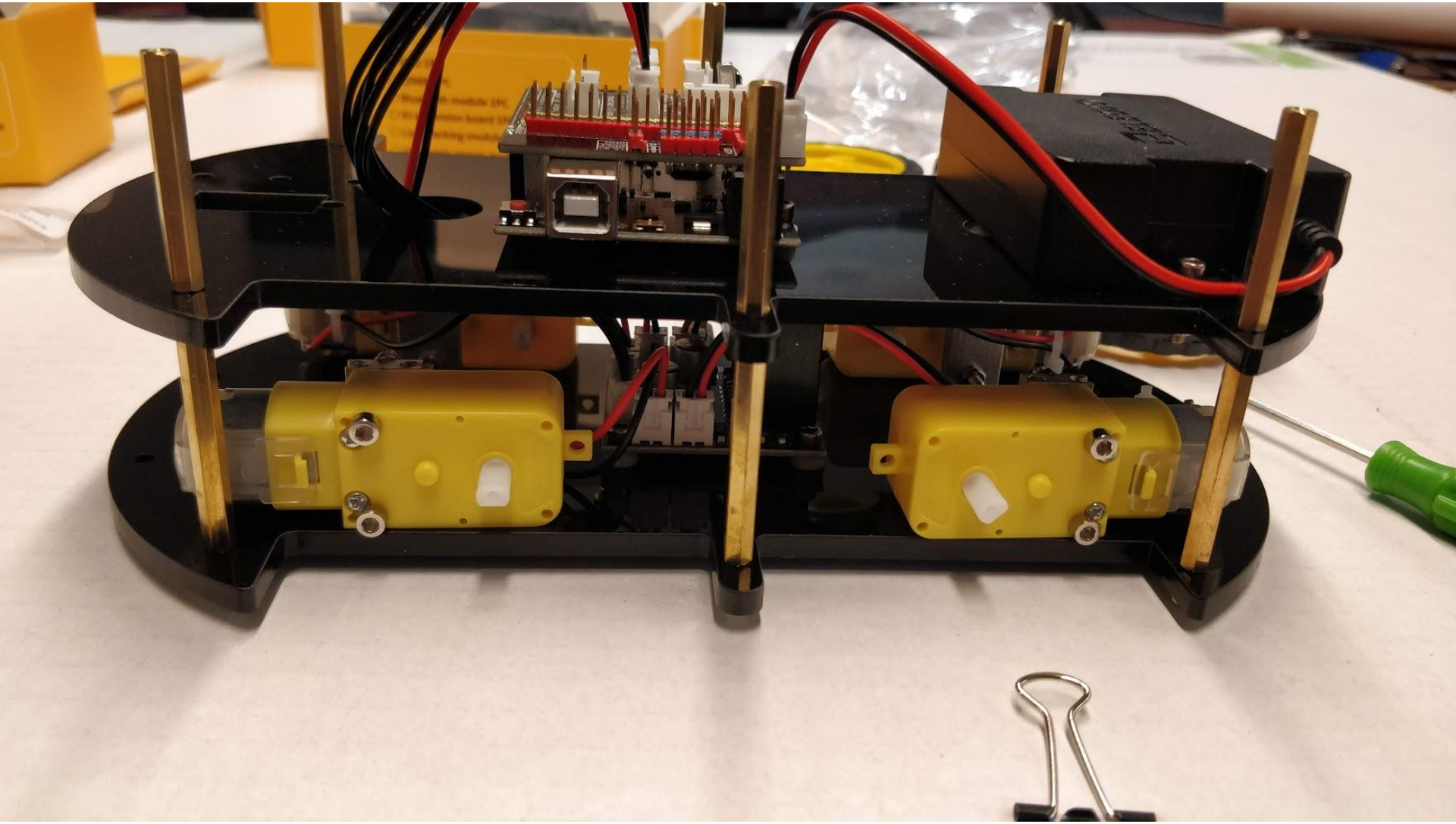


# Building the RC Car

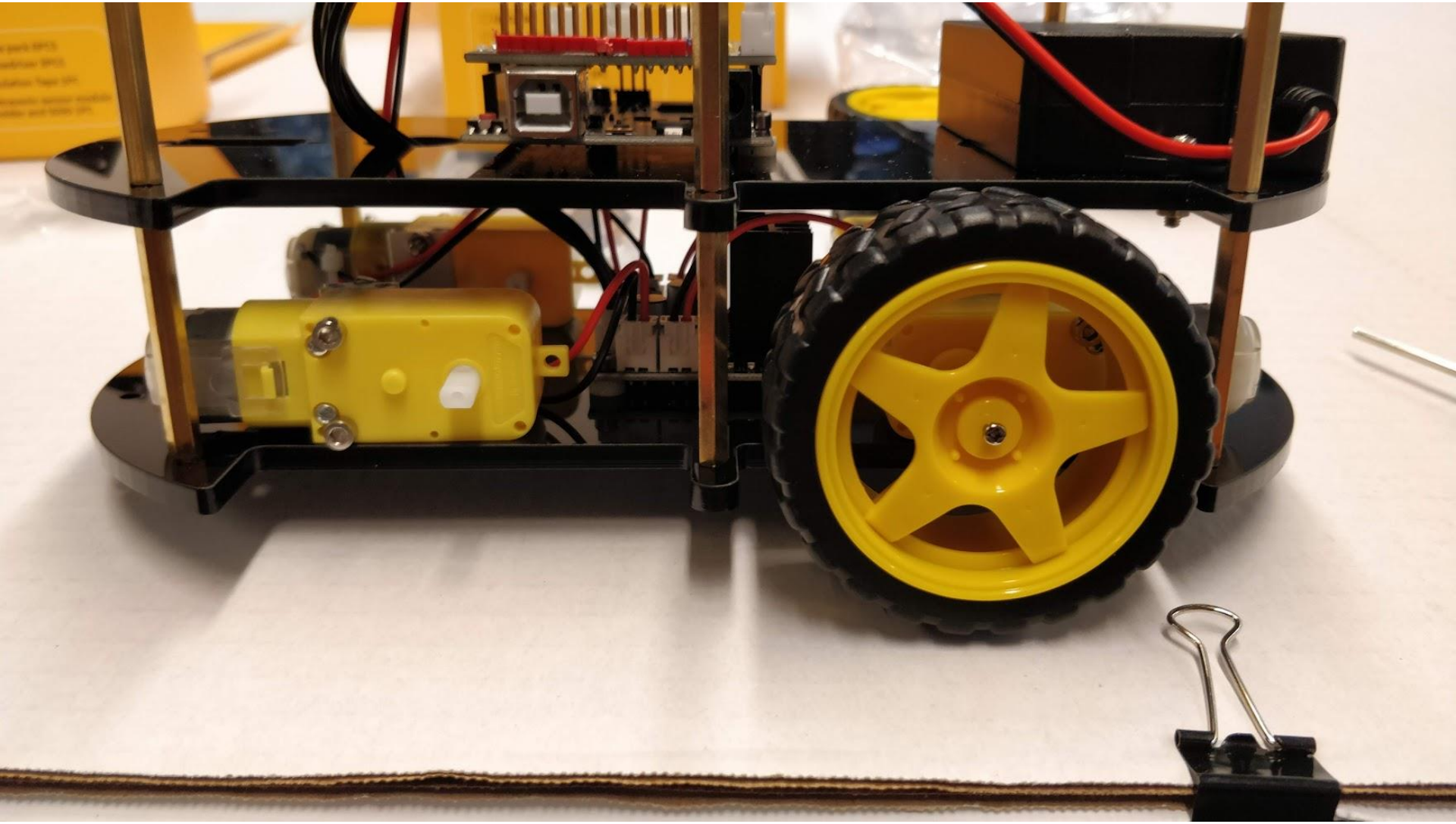
# Building the RC Car

# Building the RC Car

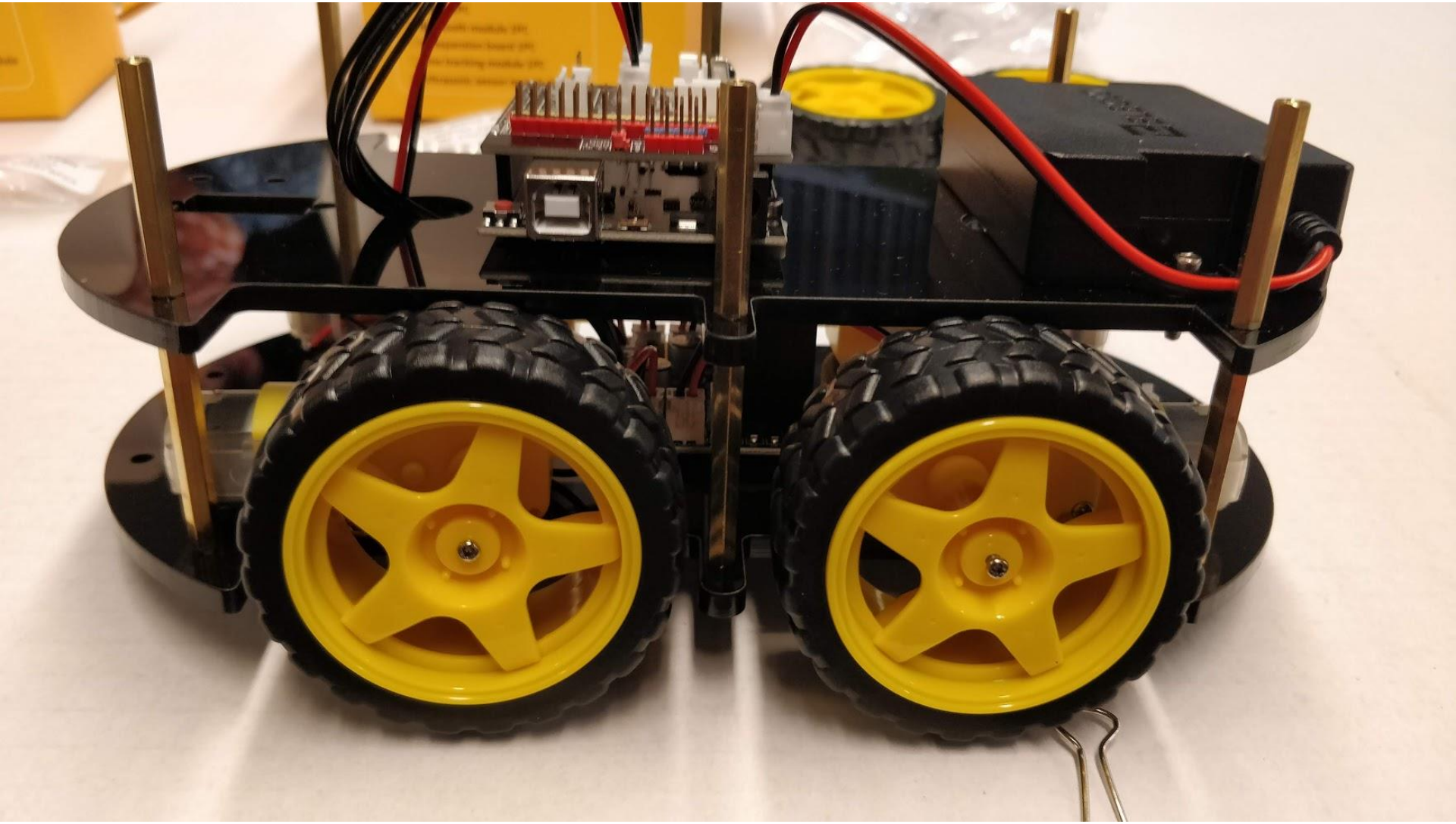
# Building the RC Car



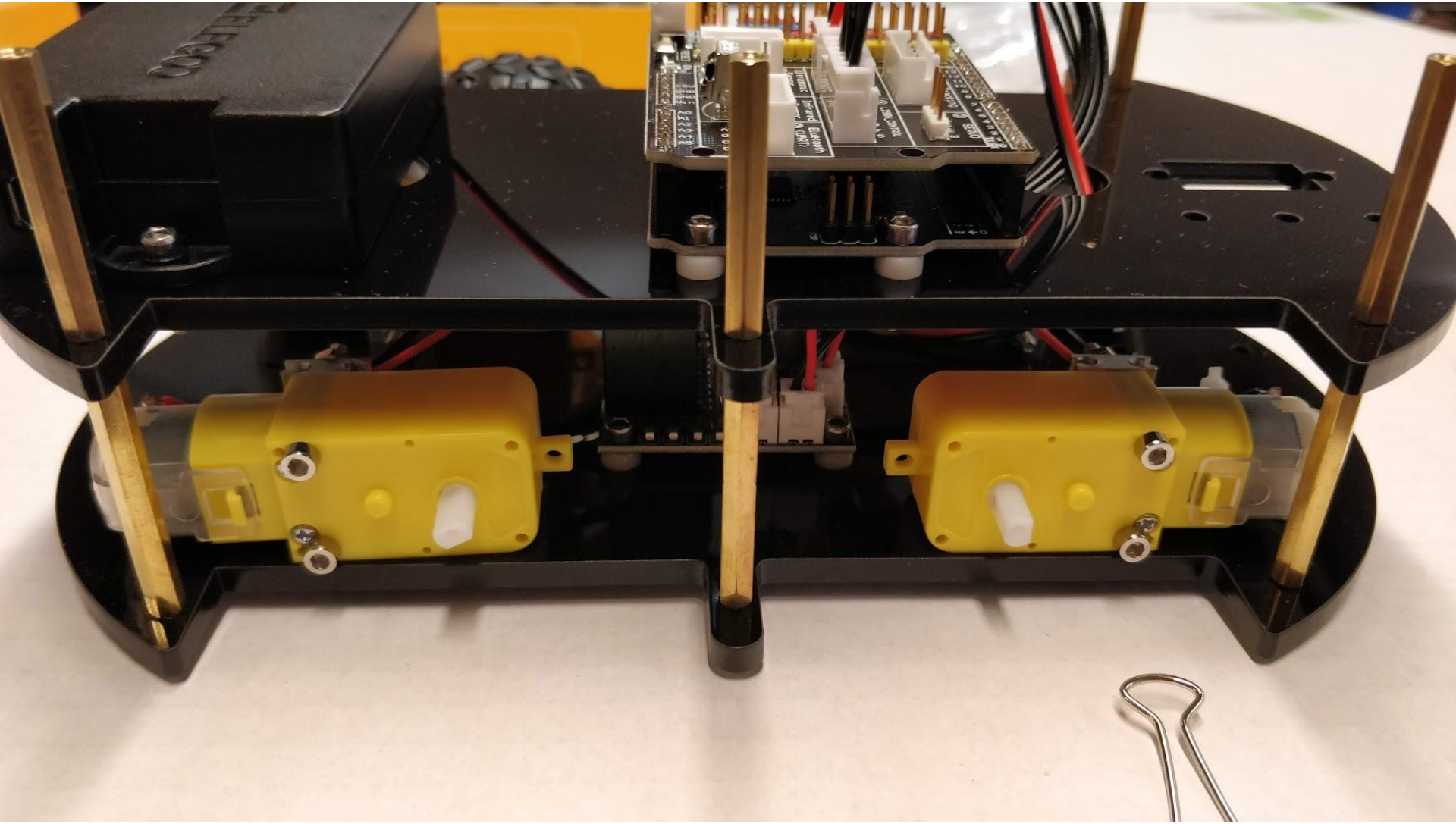
# Building the RC Car



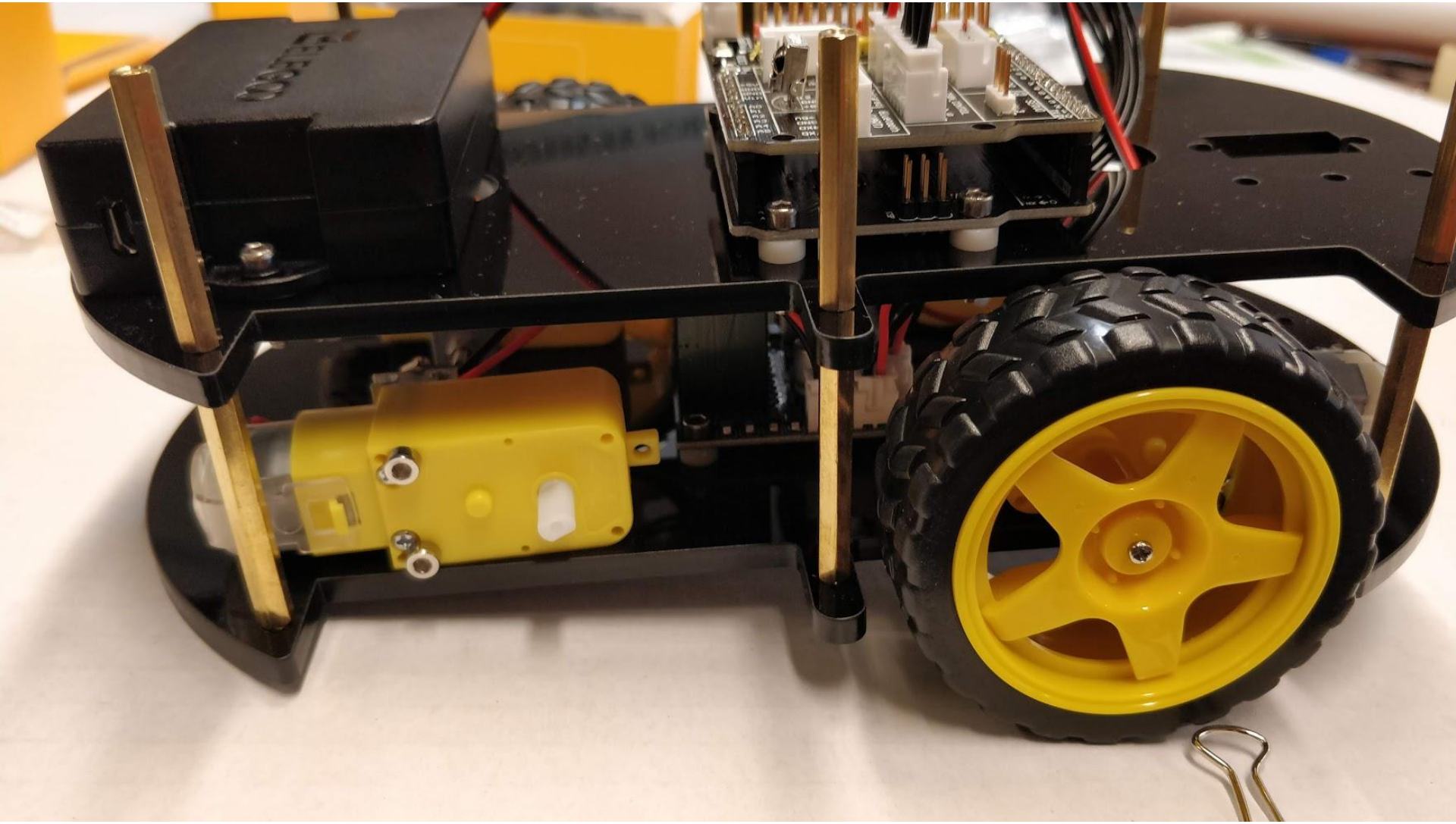
# Building the RC Car



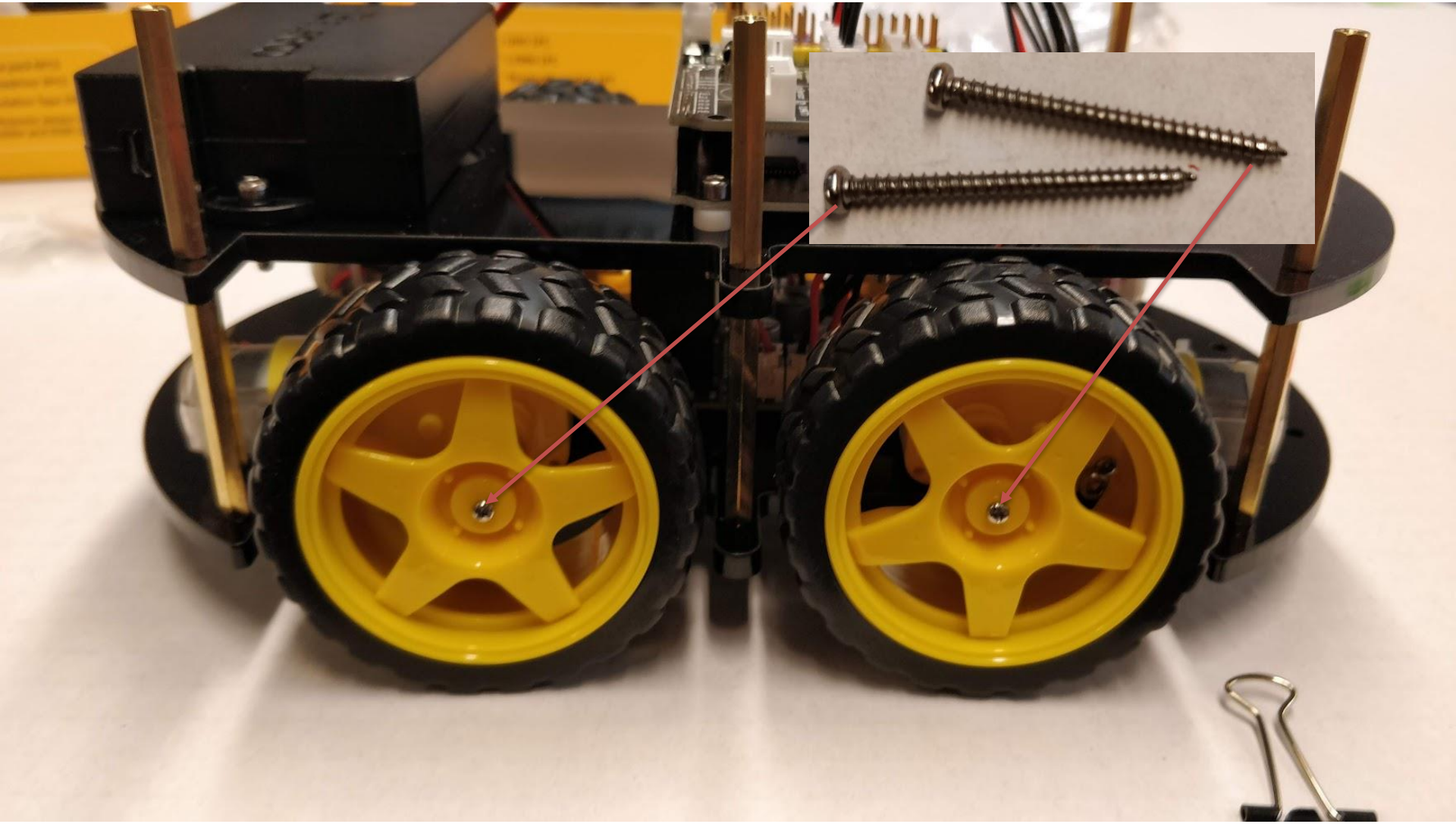
# Building the RC Car



# Building the RC Car

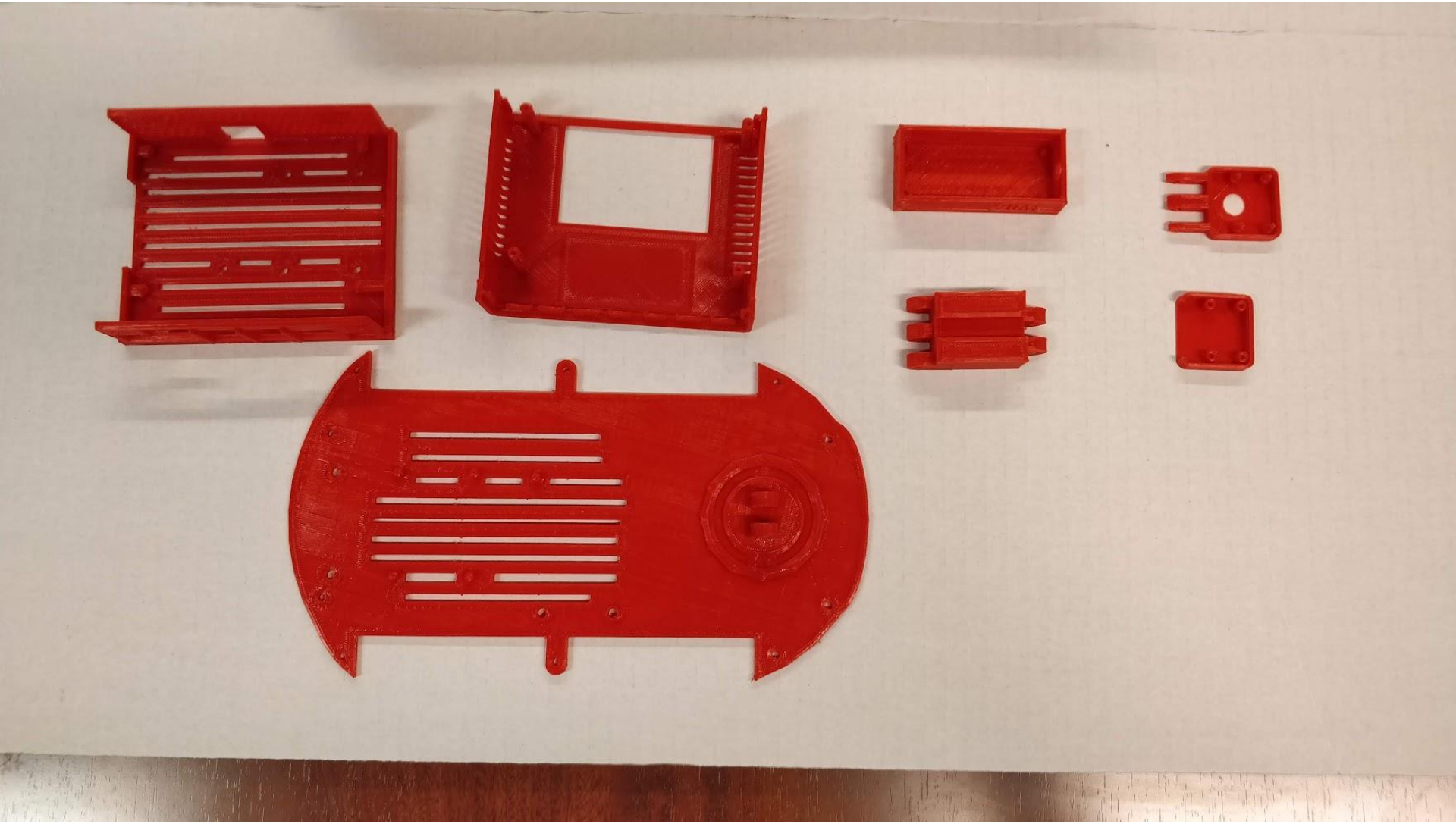


# Building the RC Car

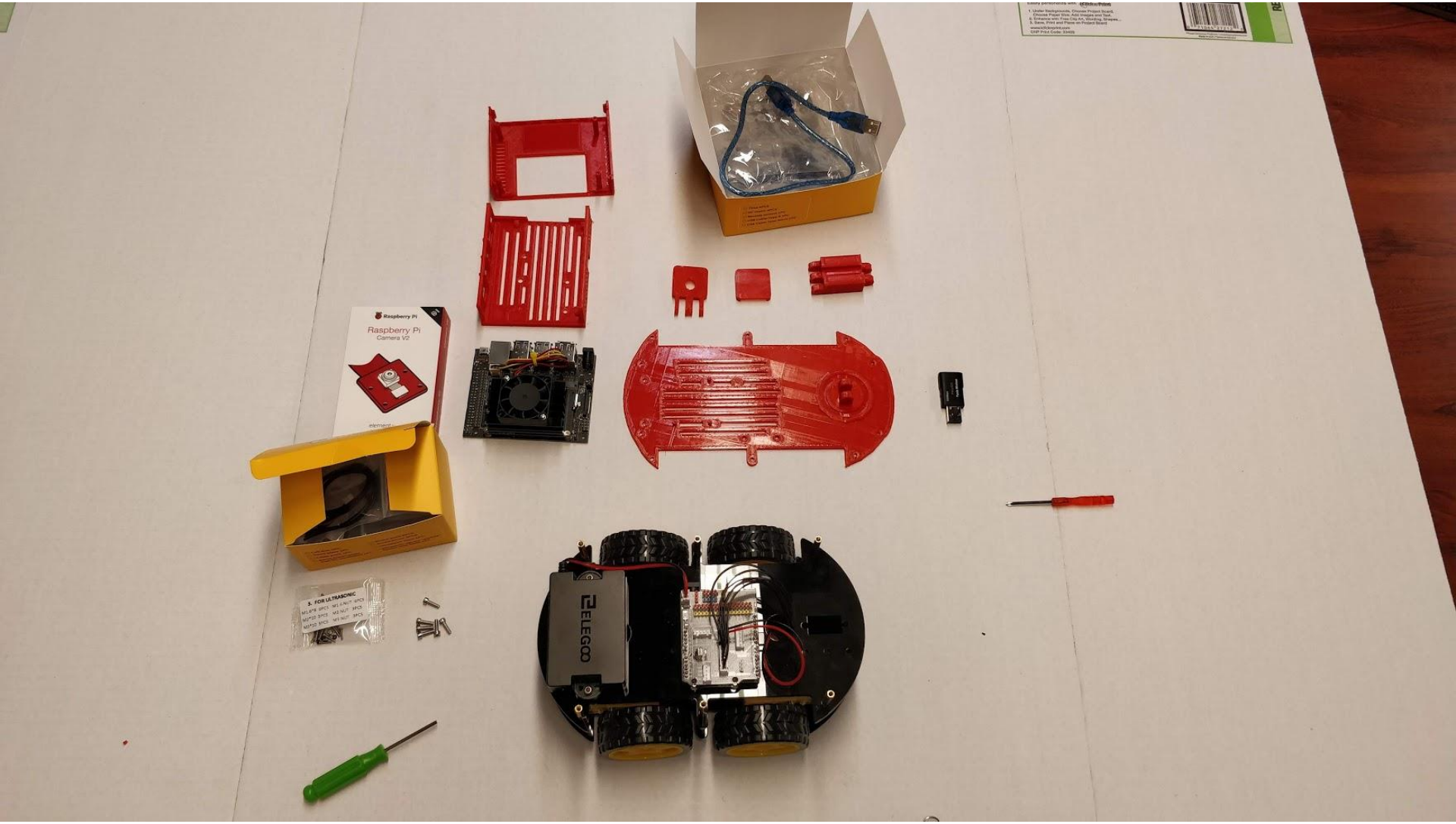


# Building the RC Car

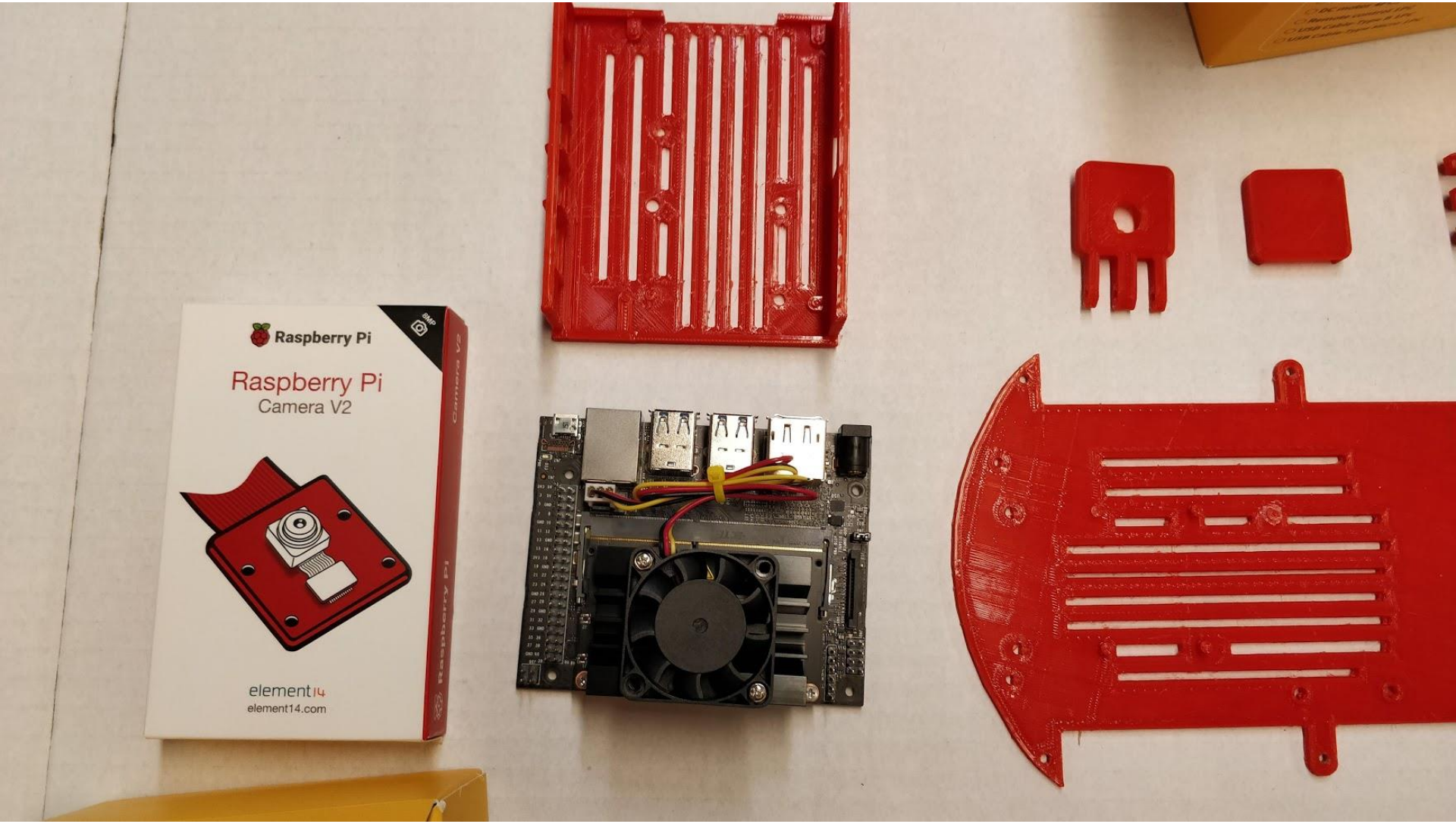




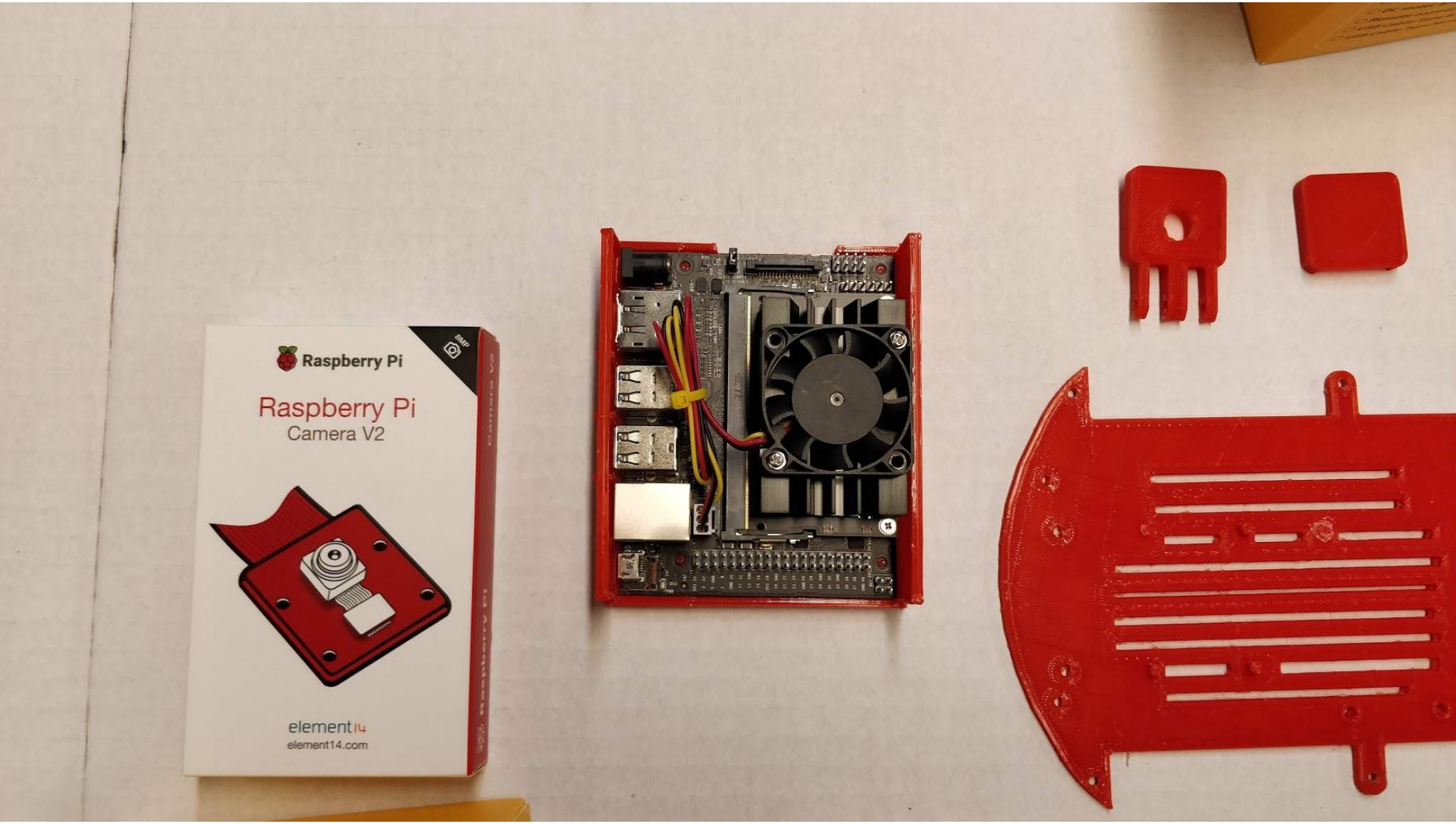
# Building the RC Car



# Building the RC Car

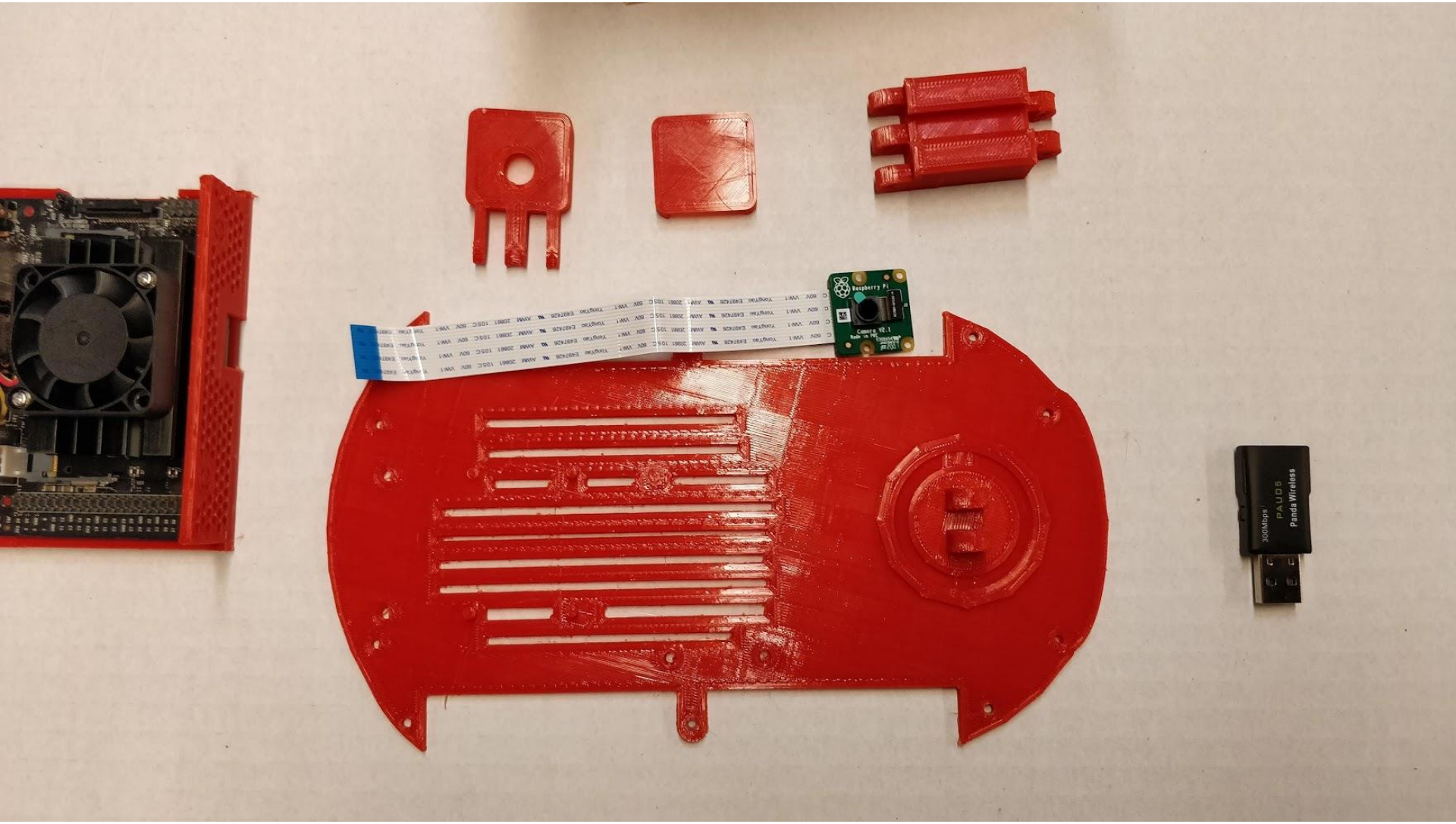


# Building the RC Car

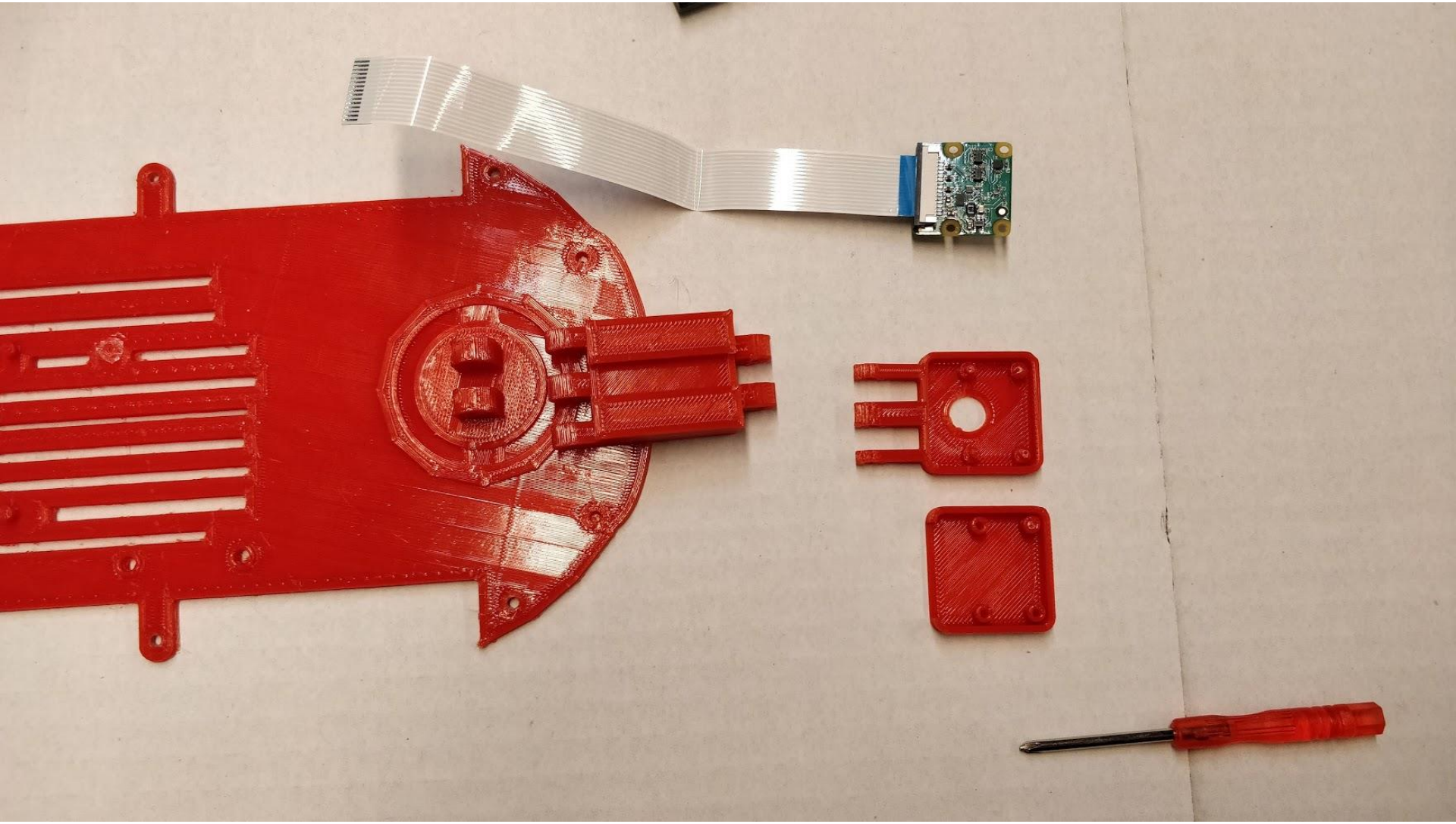


# Building the RC Car

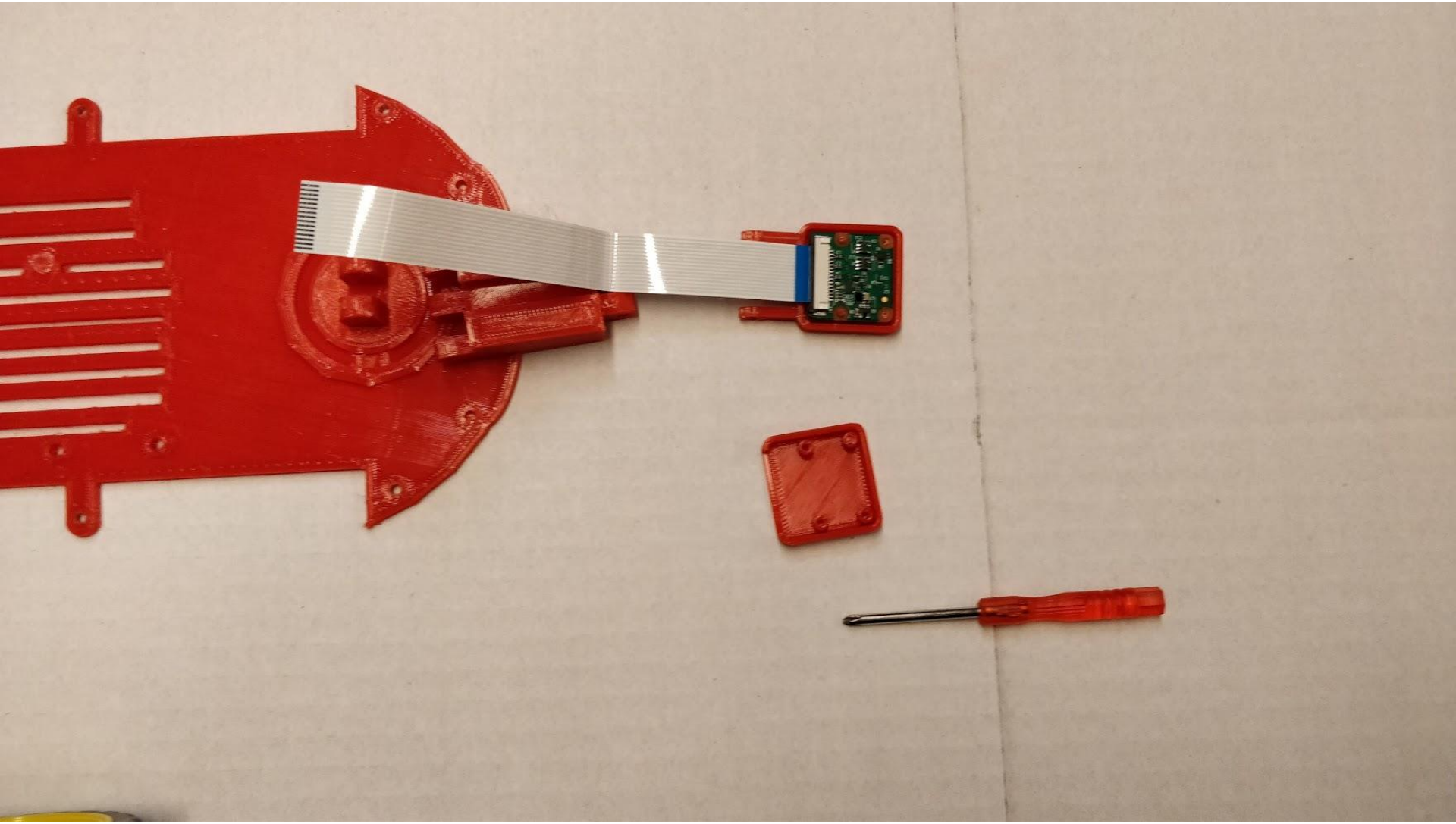
# Building the RC Car



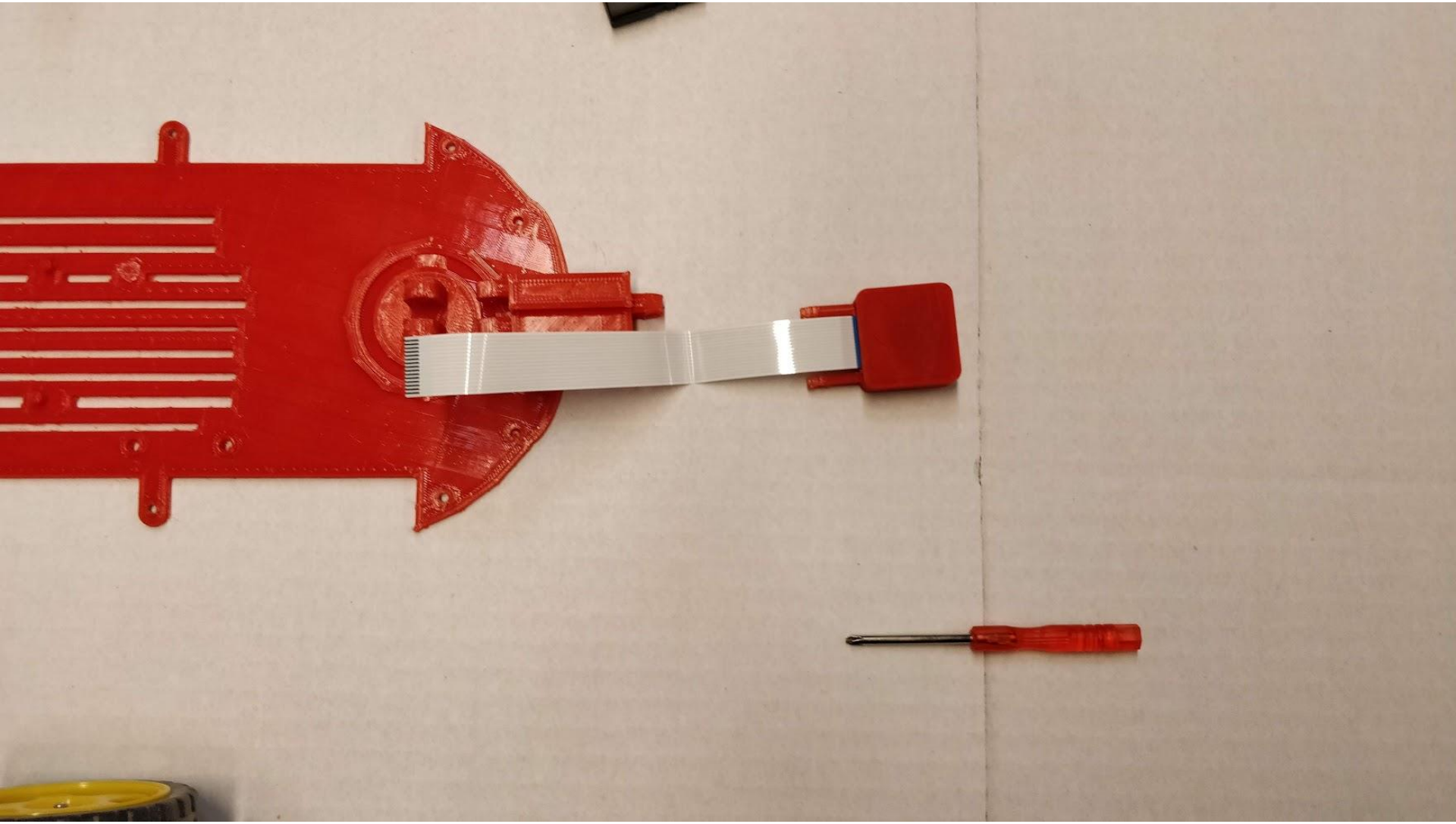
# Building the RC Car

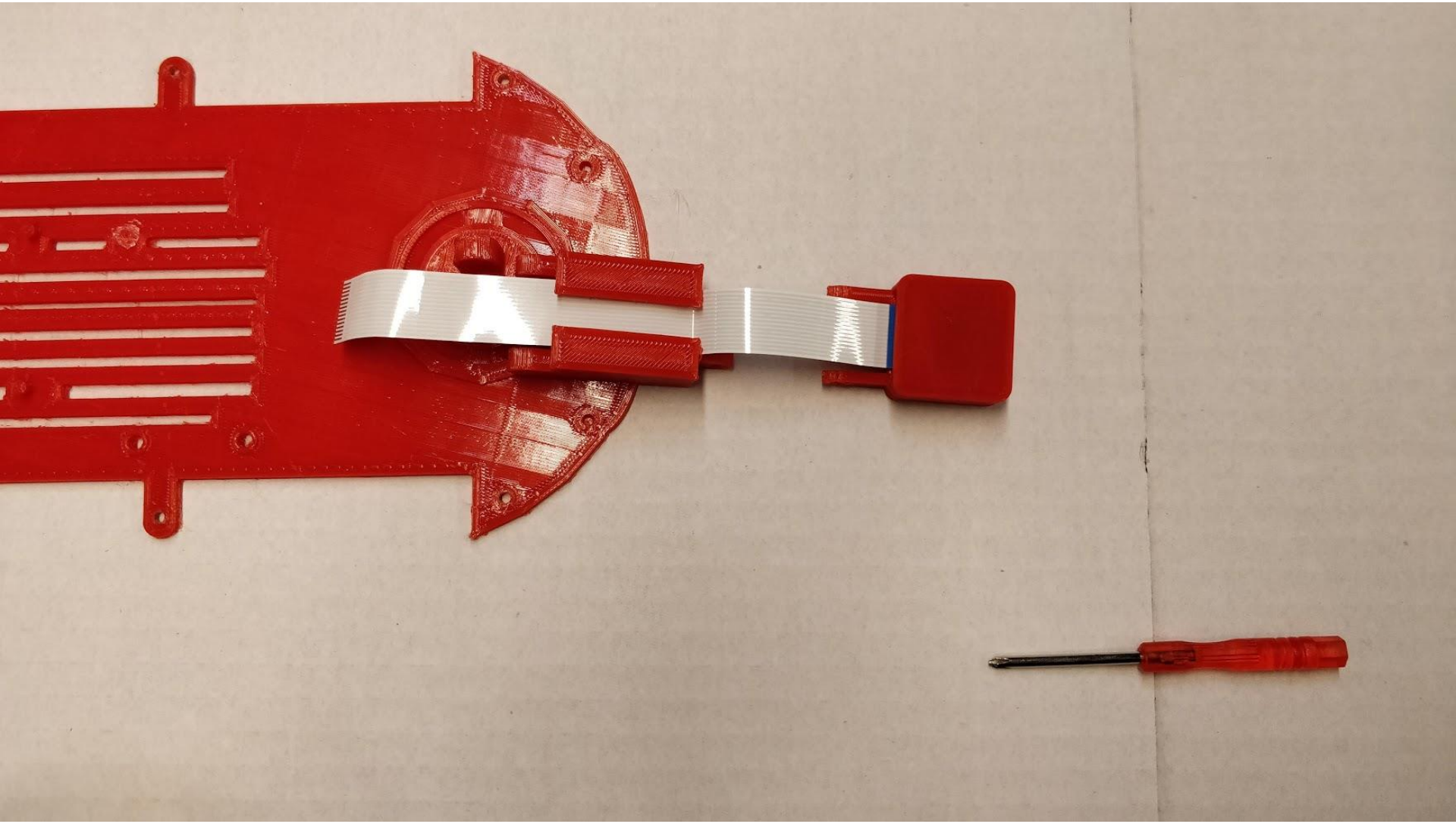


# Building the RC Car

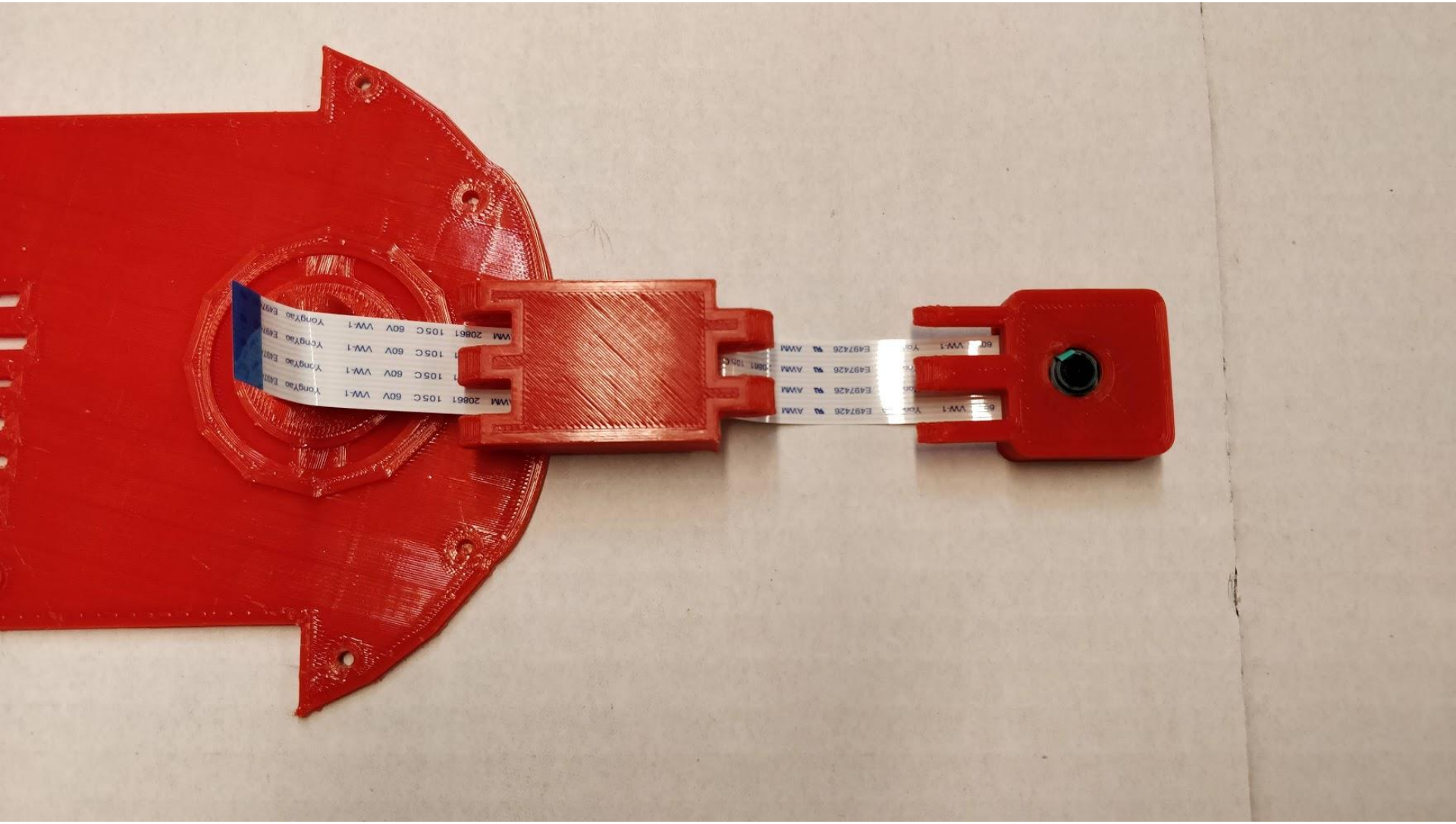


# Building the RC Car

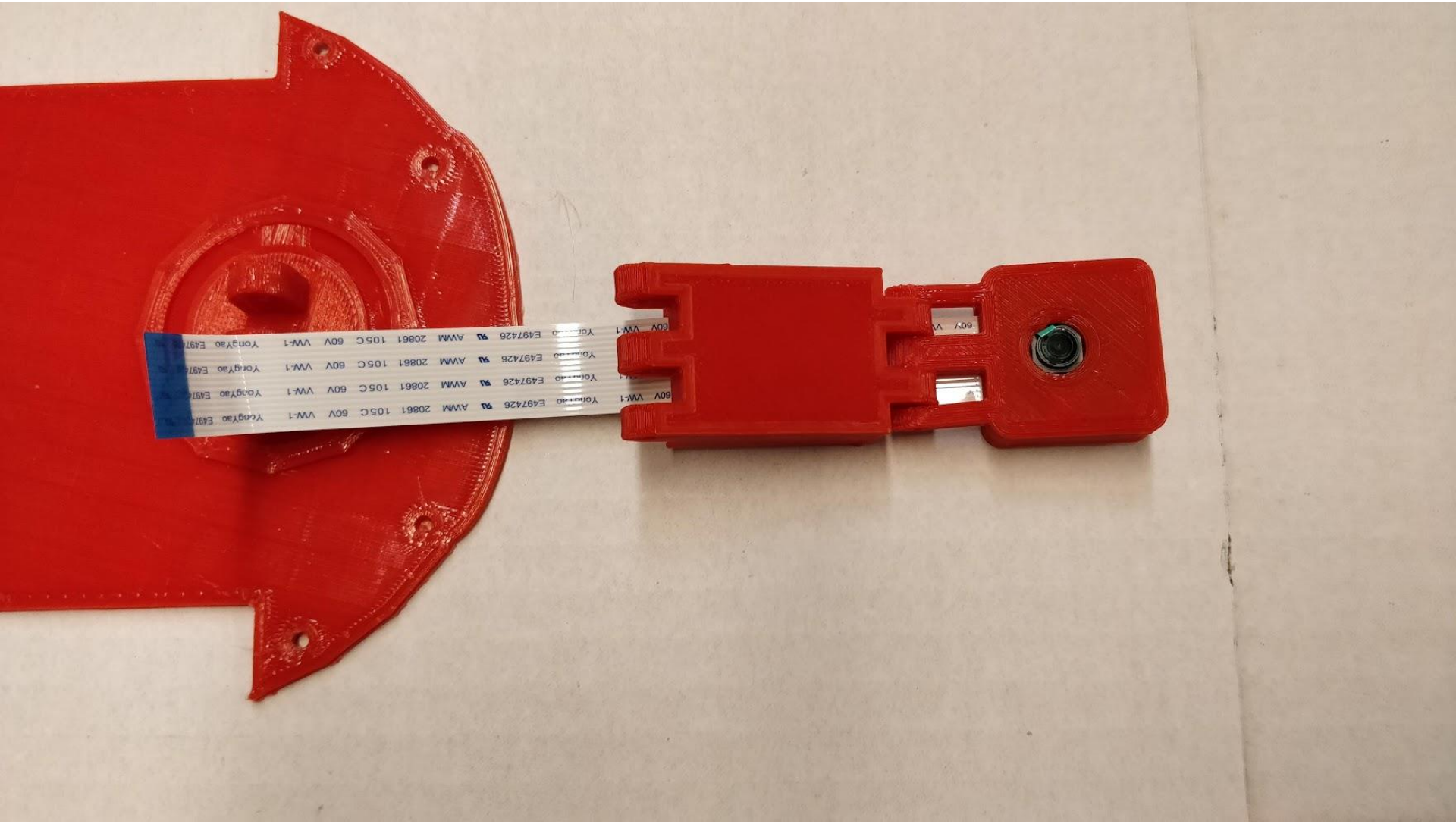




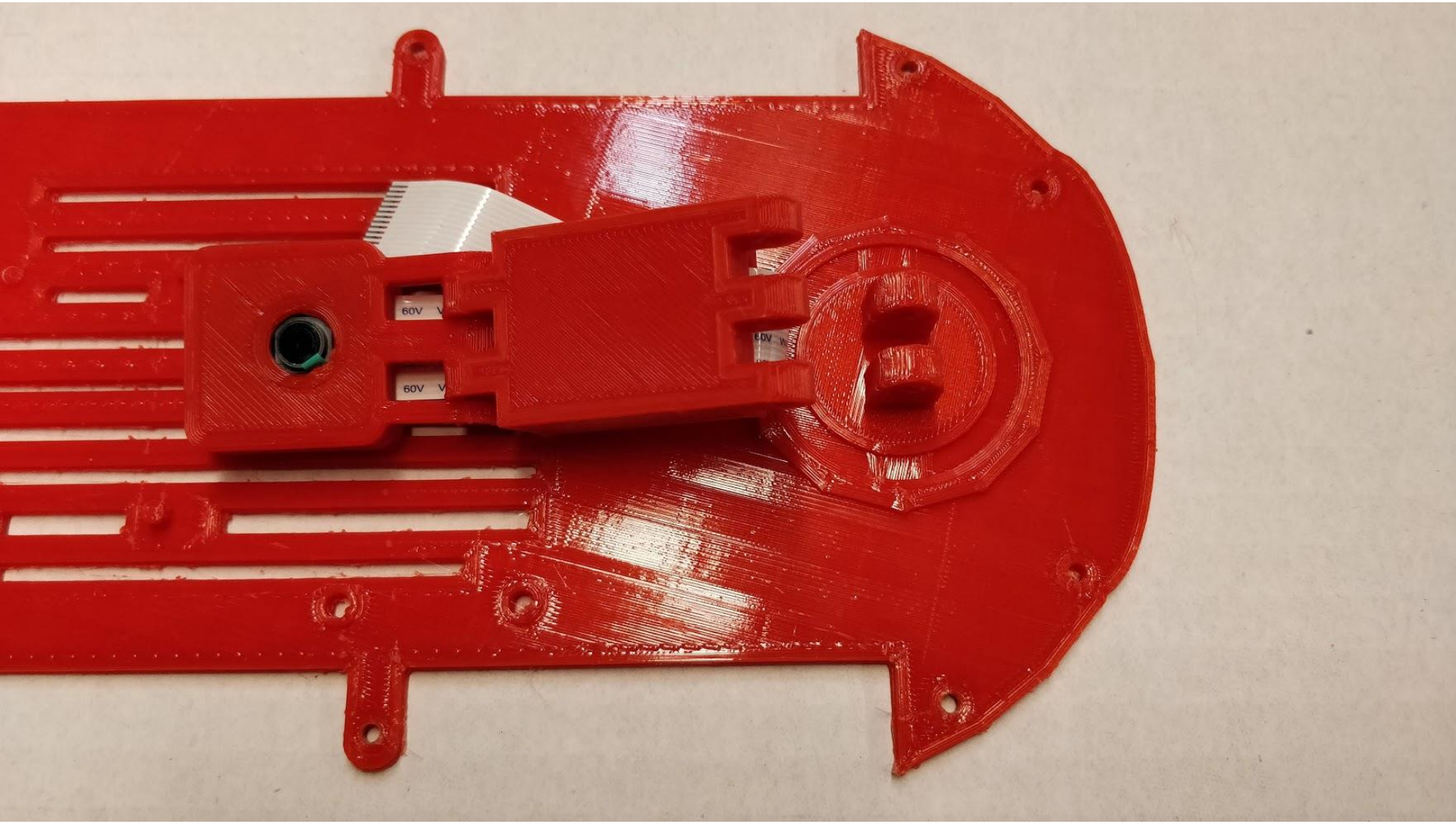
# Building the RC Car



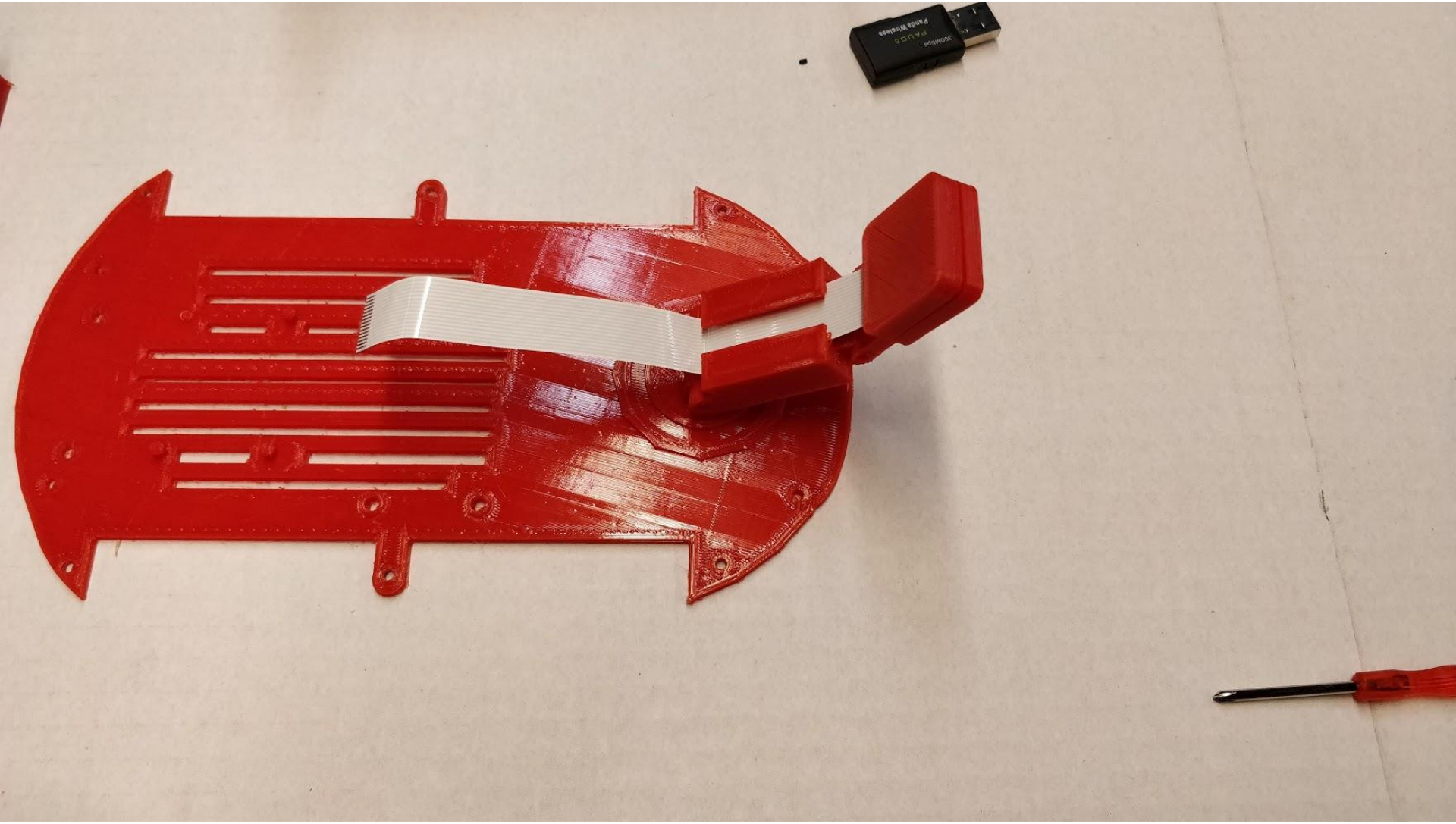
# Building the RC Car



# Building the RC Car



# Building the RC Car



# Building the RC Car

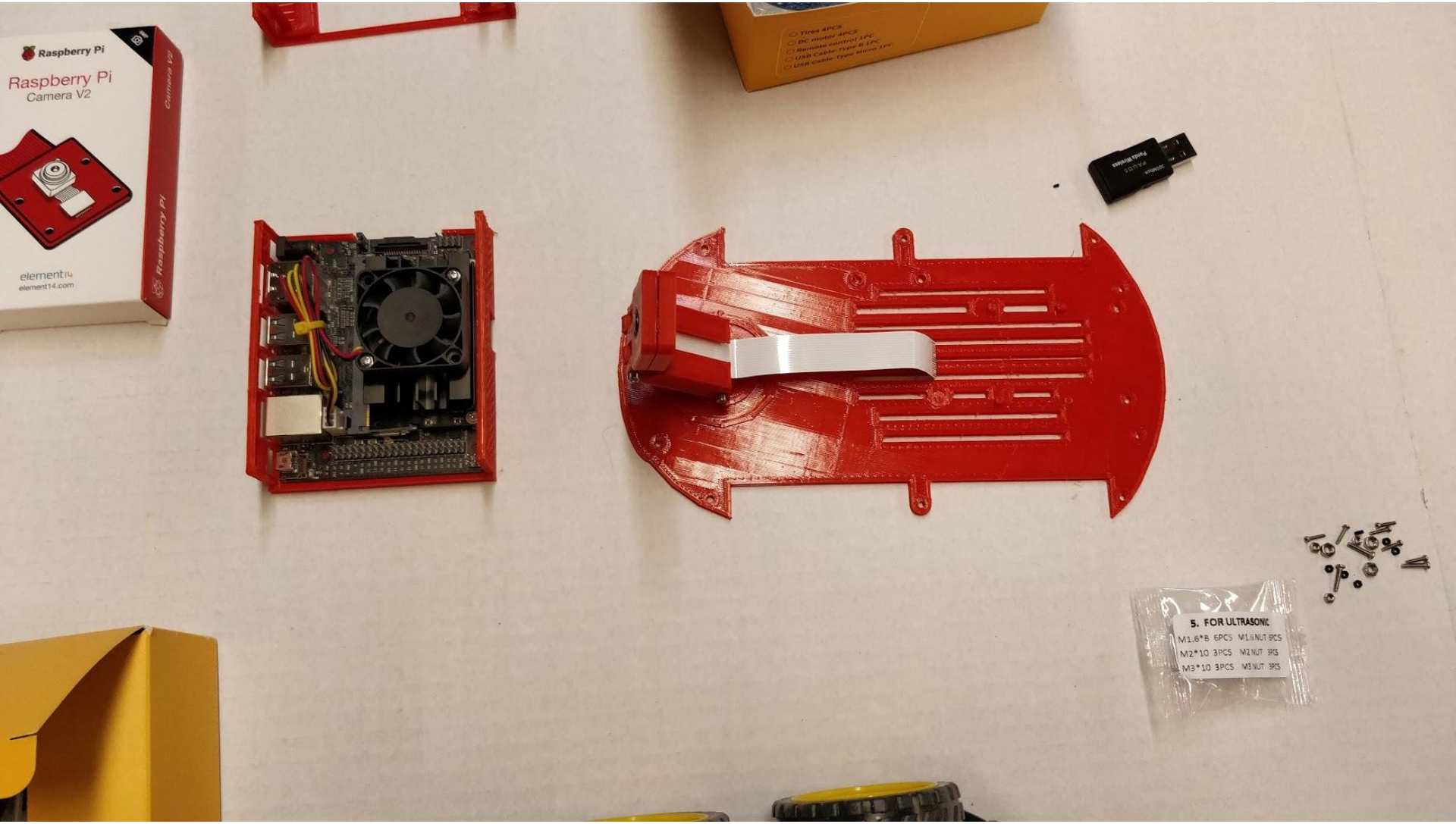
# Building the RC Car

# Building the RC Car

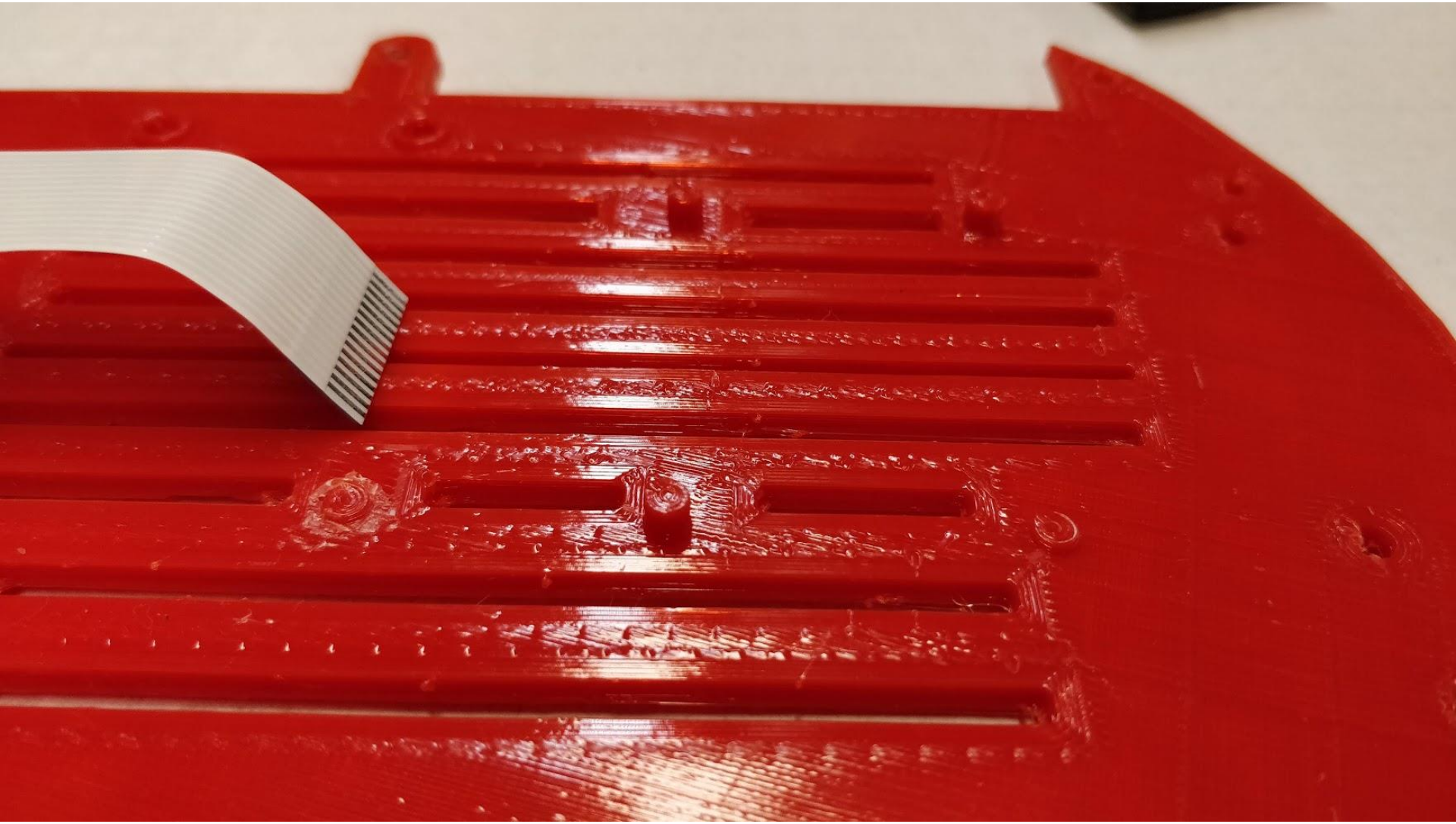
# Building the RC Car

# Building the RC Car

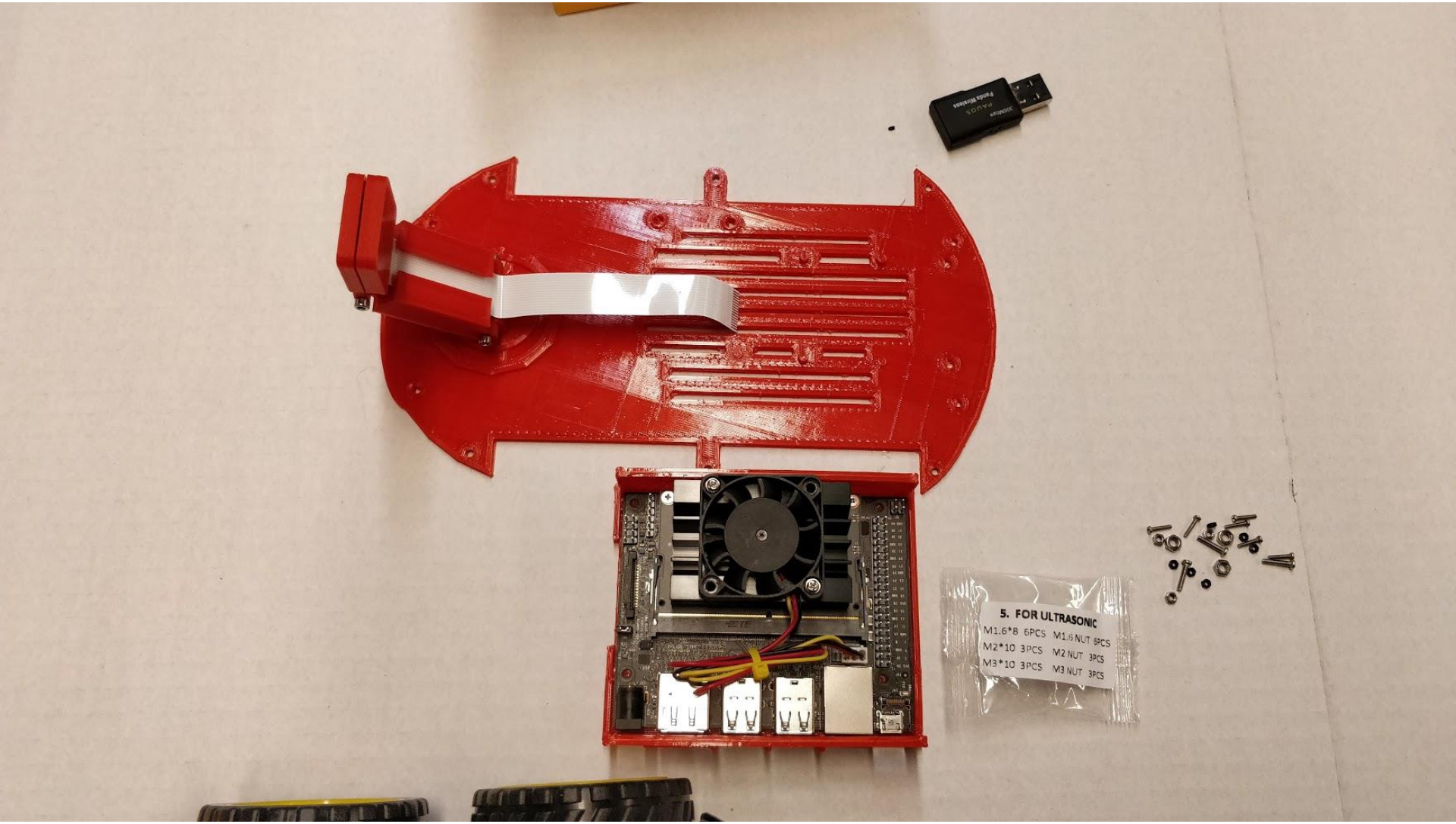
# Building the RC Car



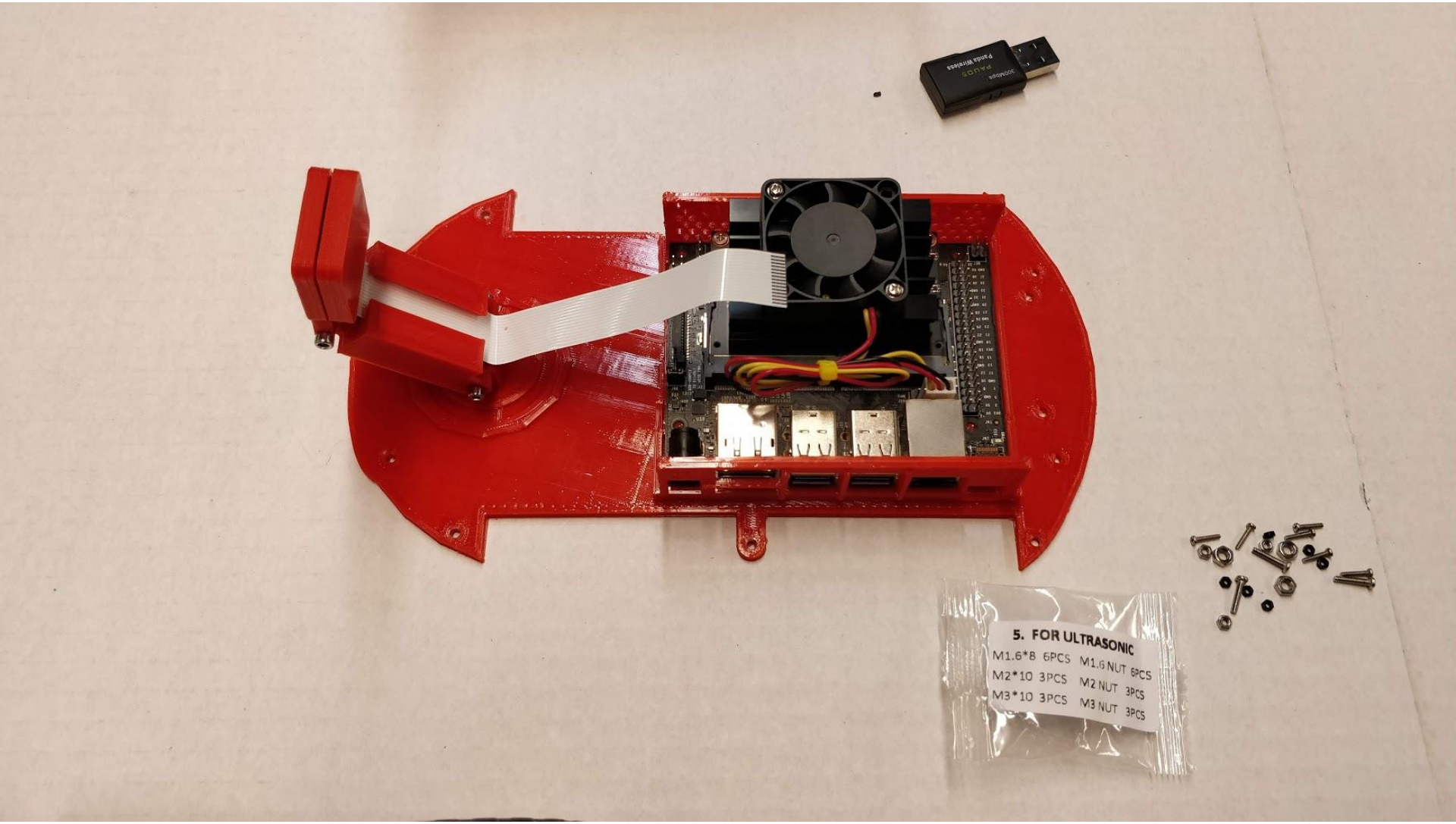
# Building the RC Car



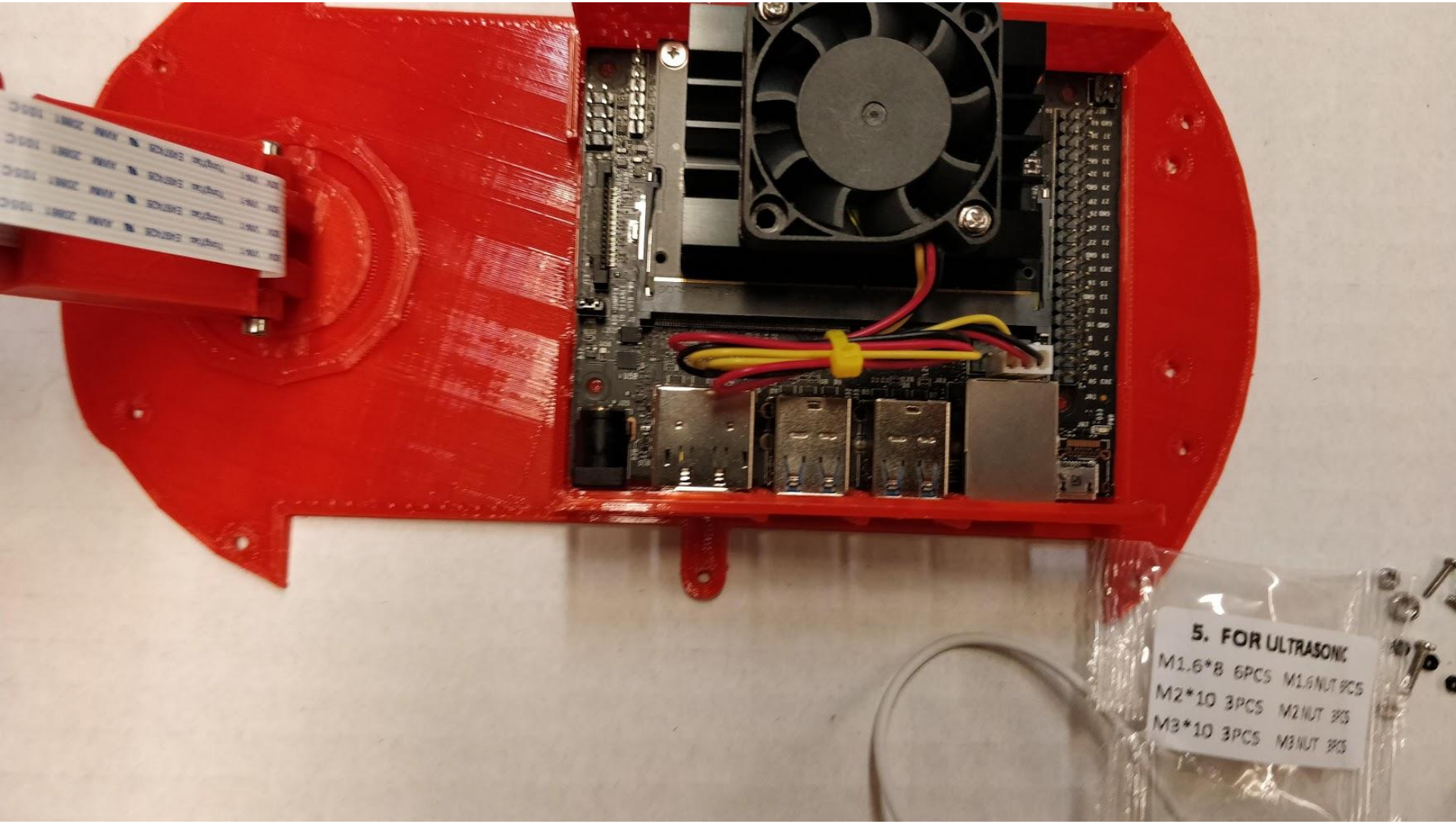
# Building the RC Car



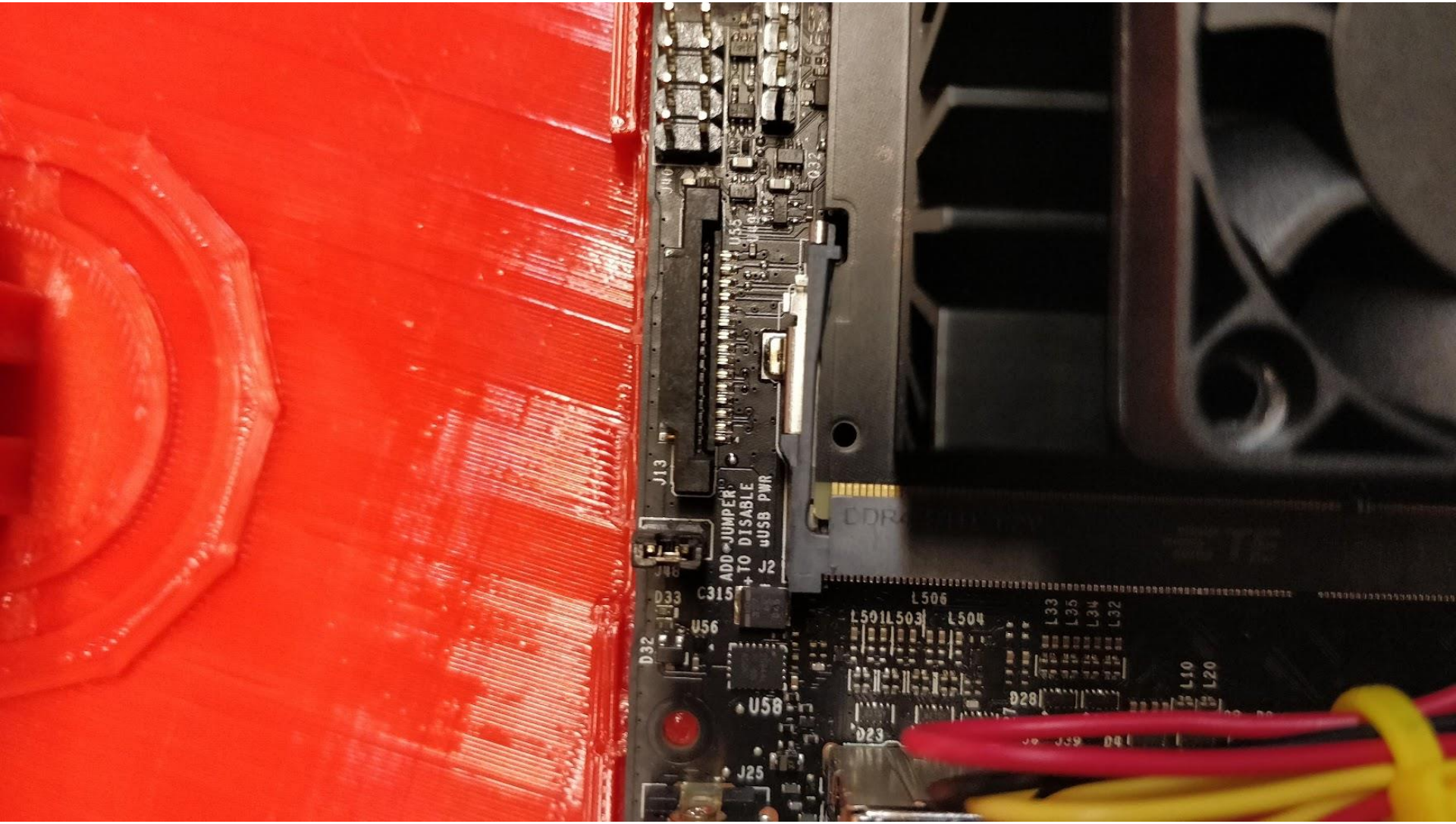
# Building the RC Car



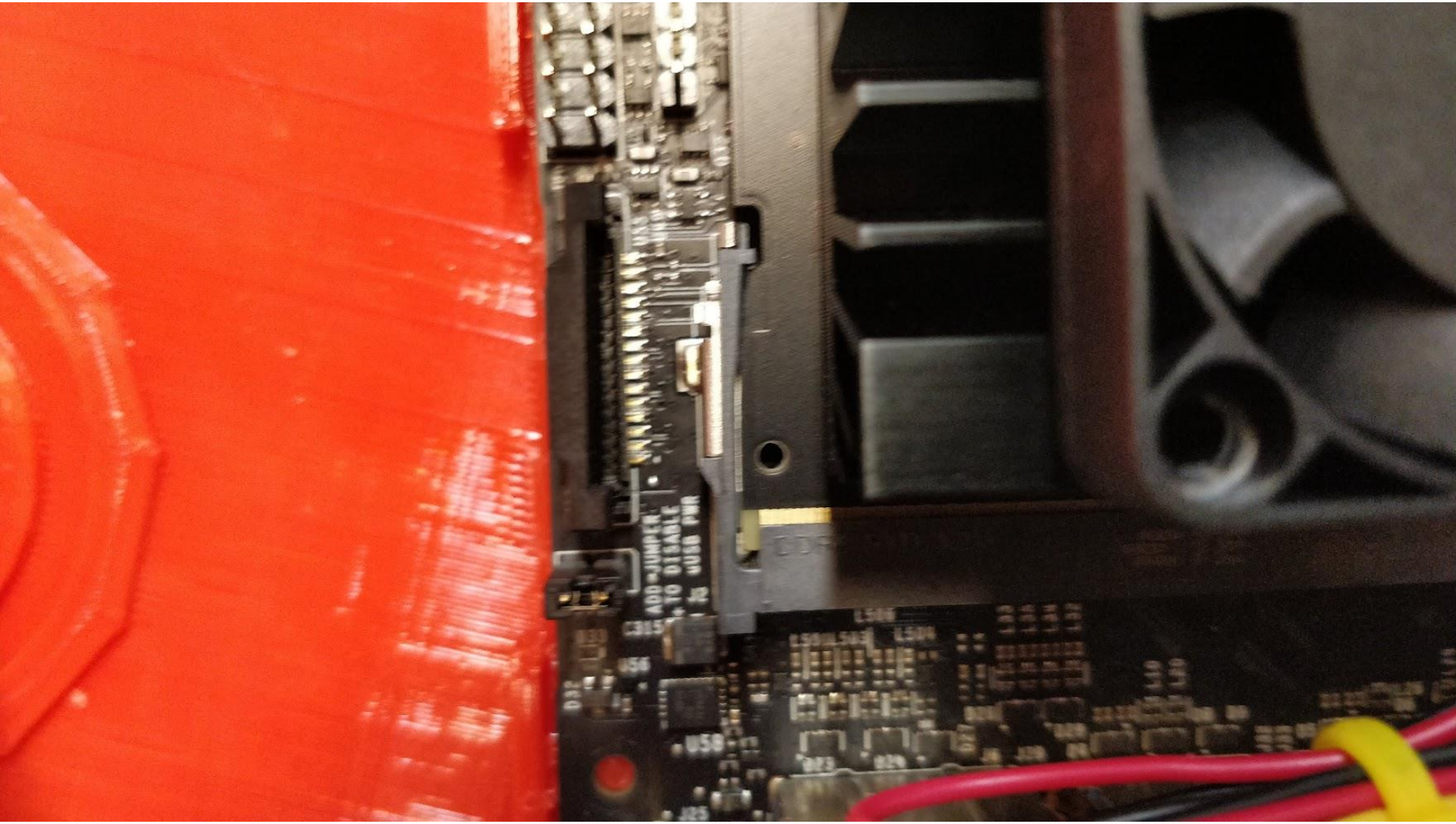
# Building the RC Car



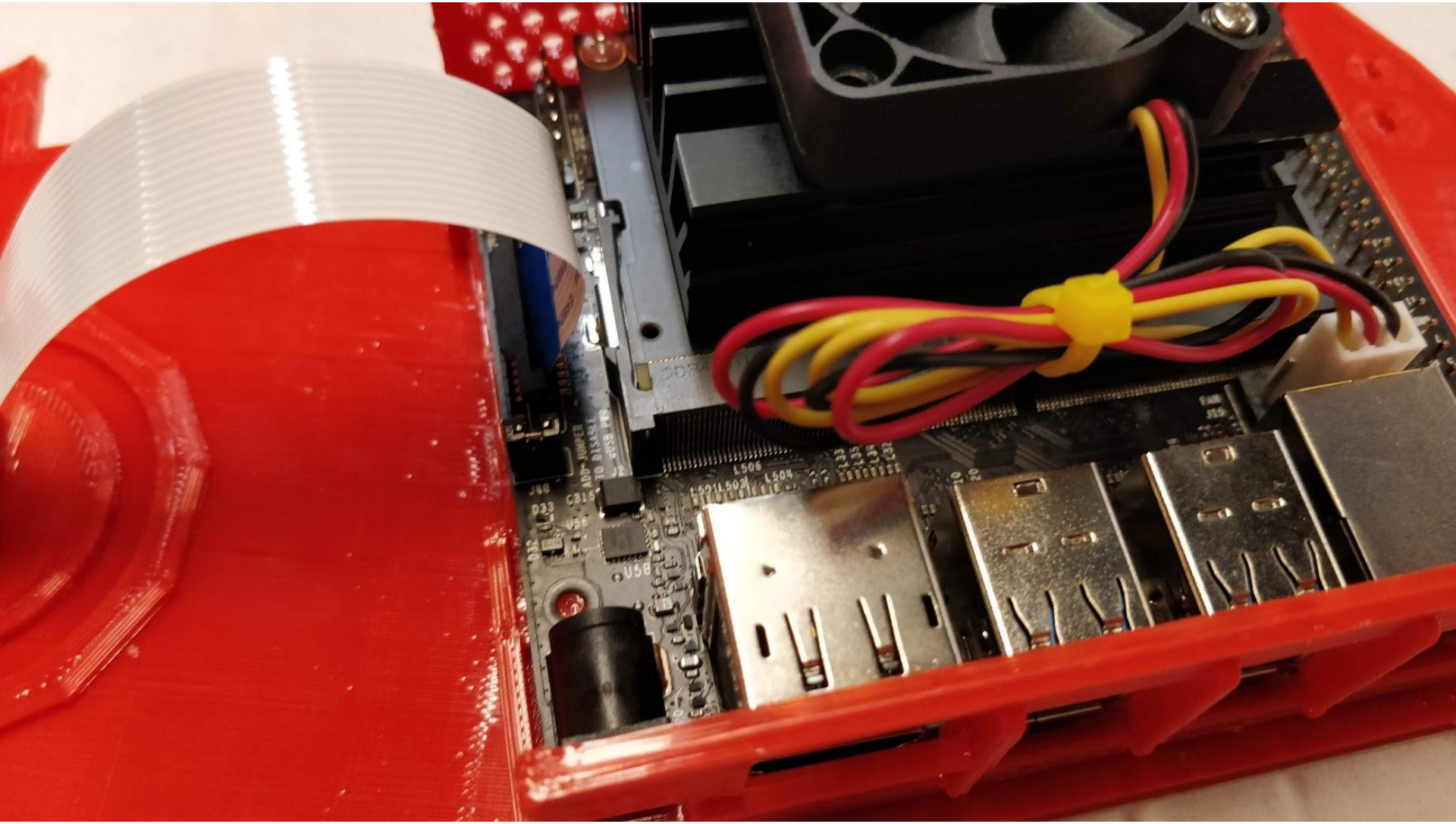
# Building the RC Car



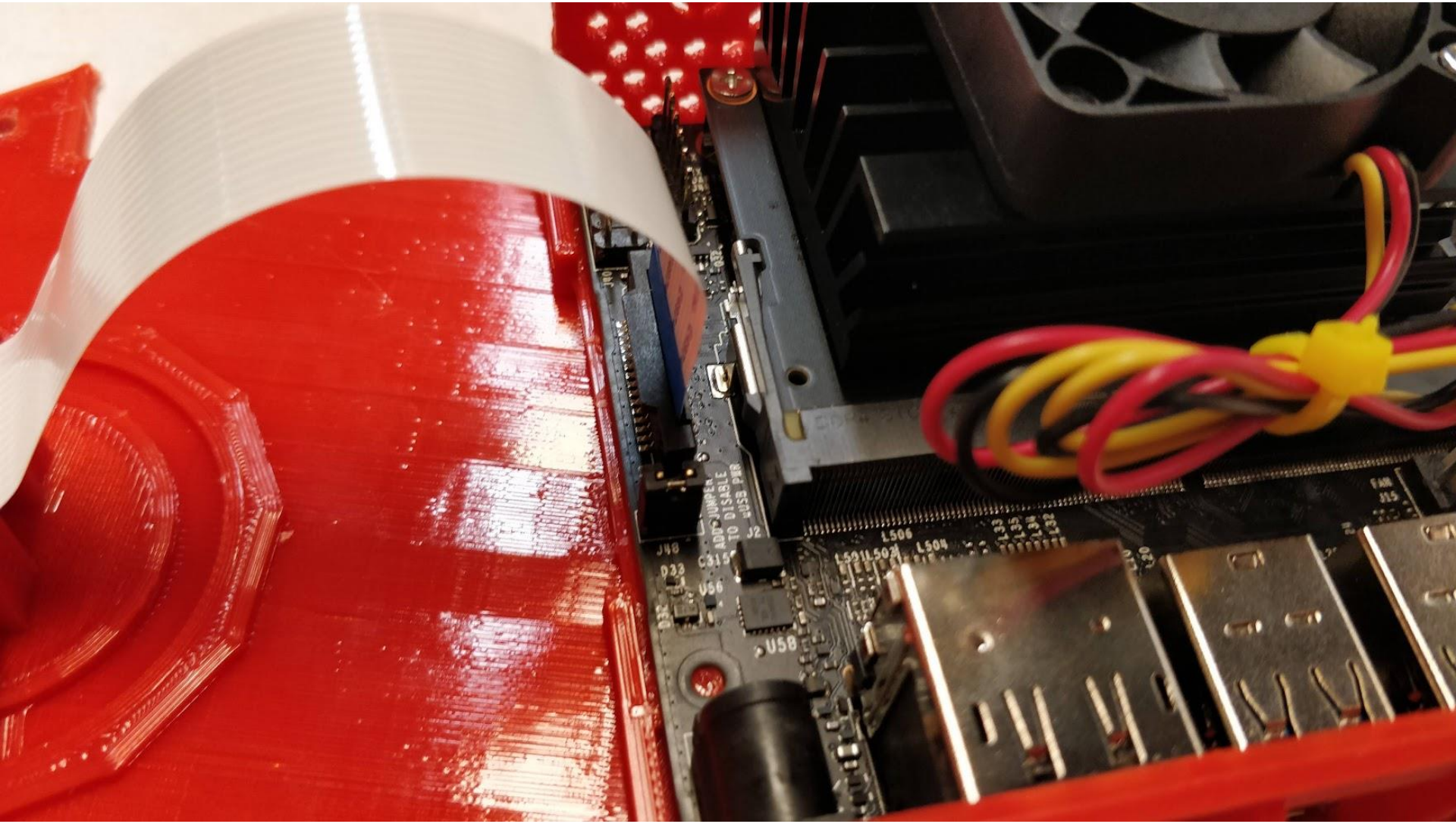
# Building the RC Car



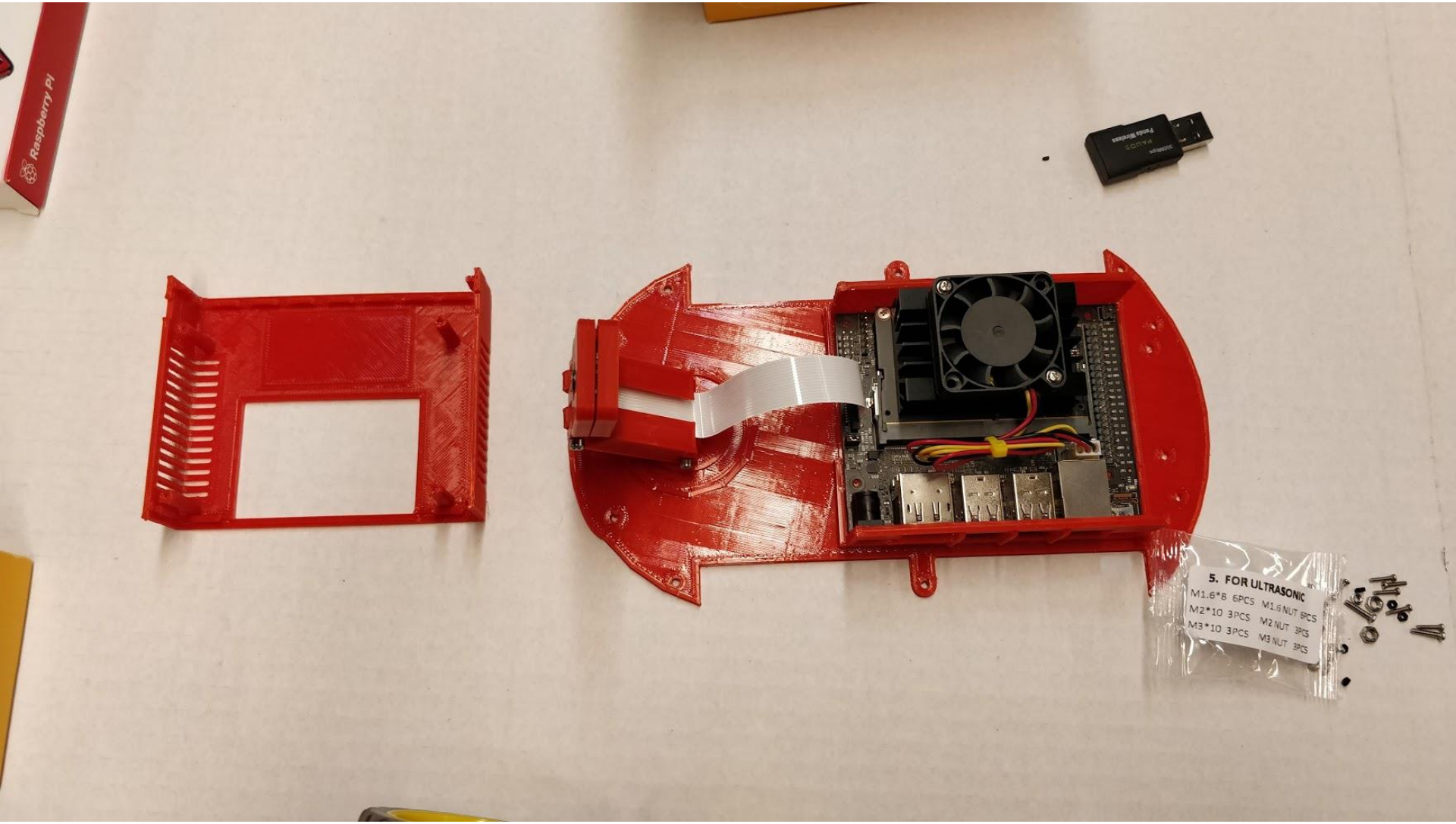
# Building the RC Car



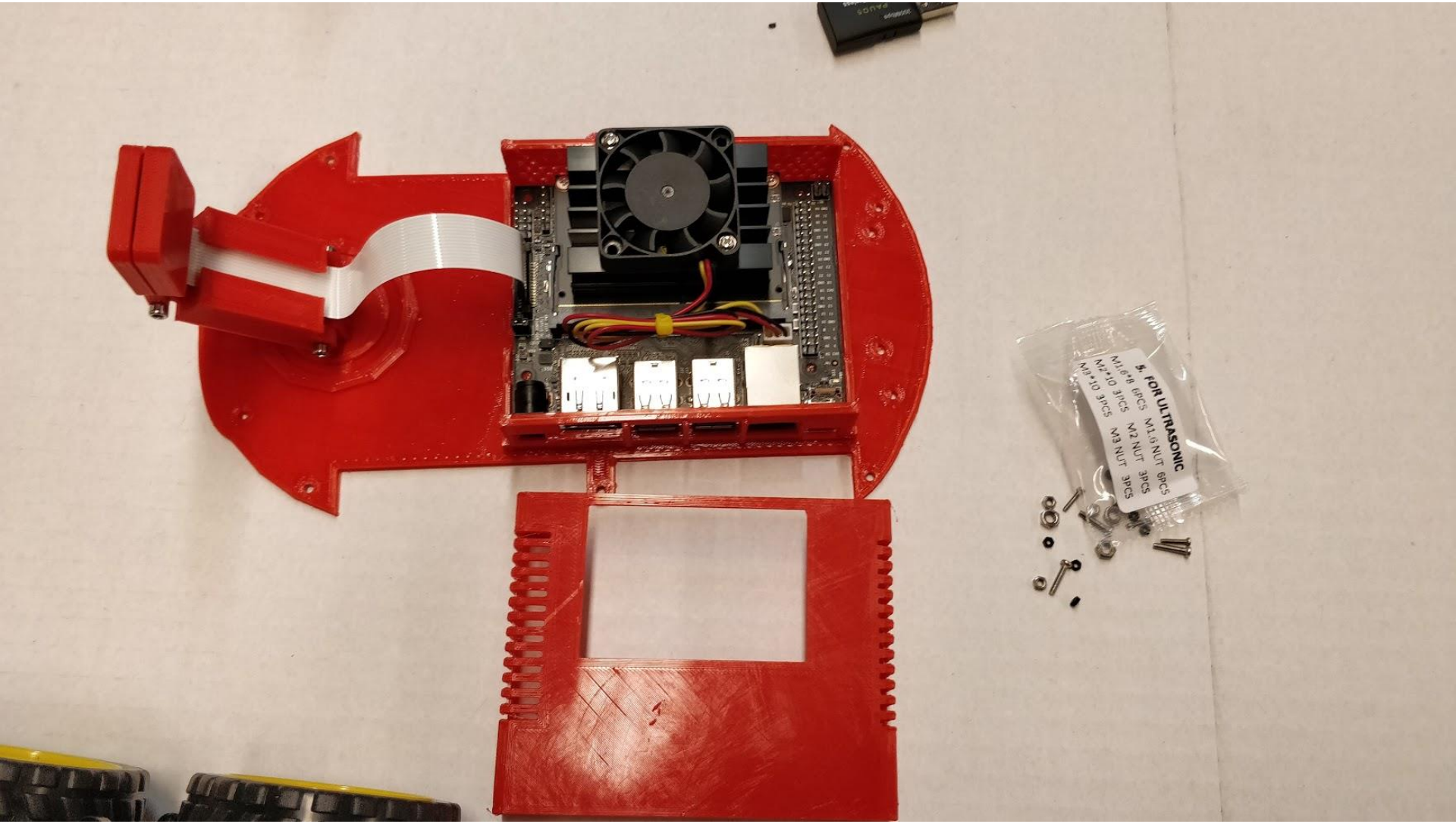
# Building the RC Car



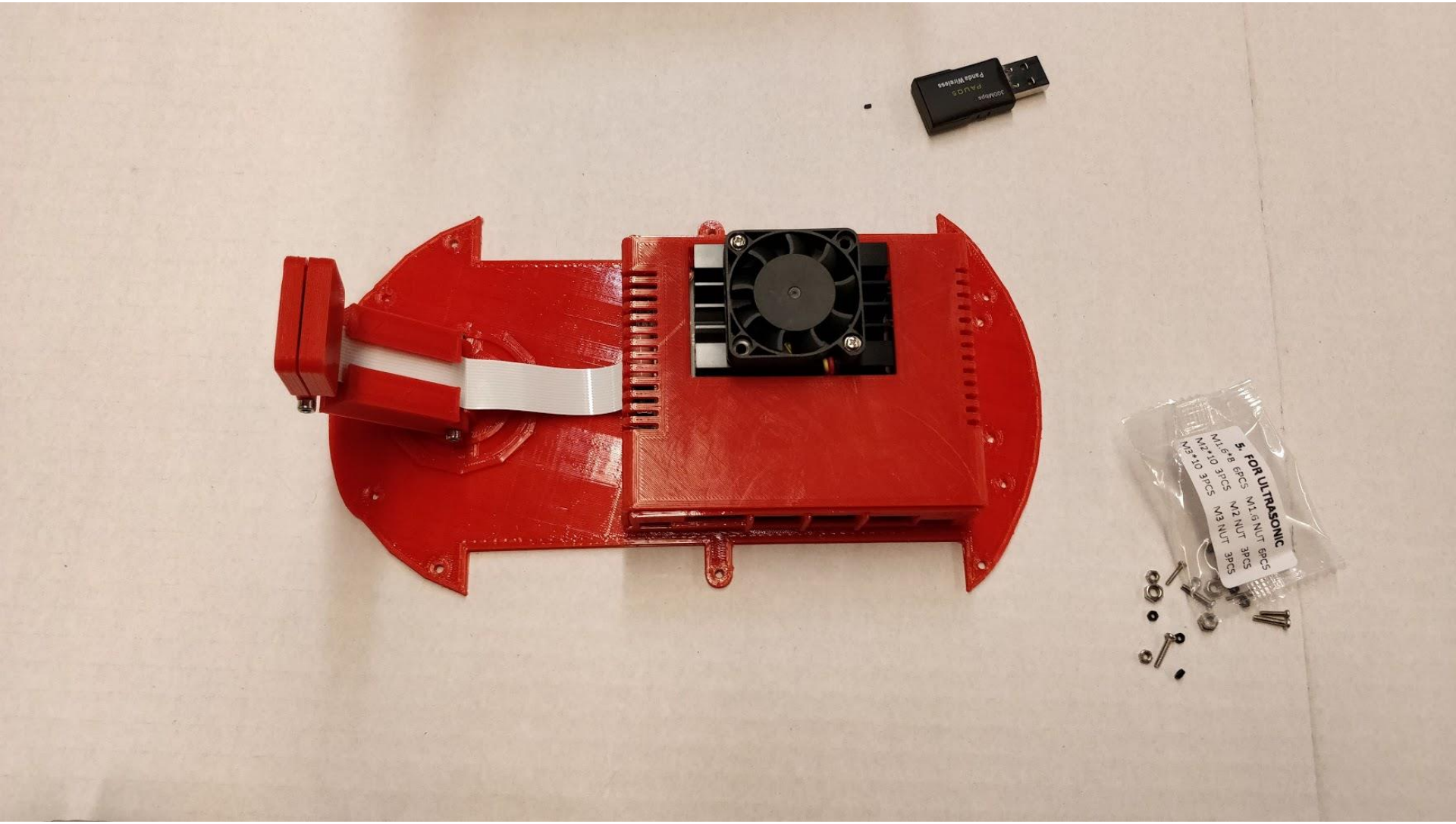
# Building the RC Car



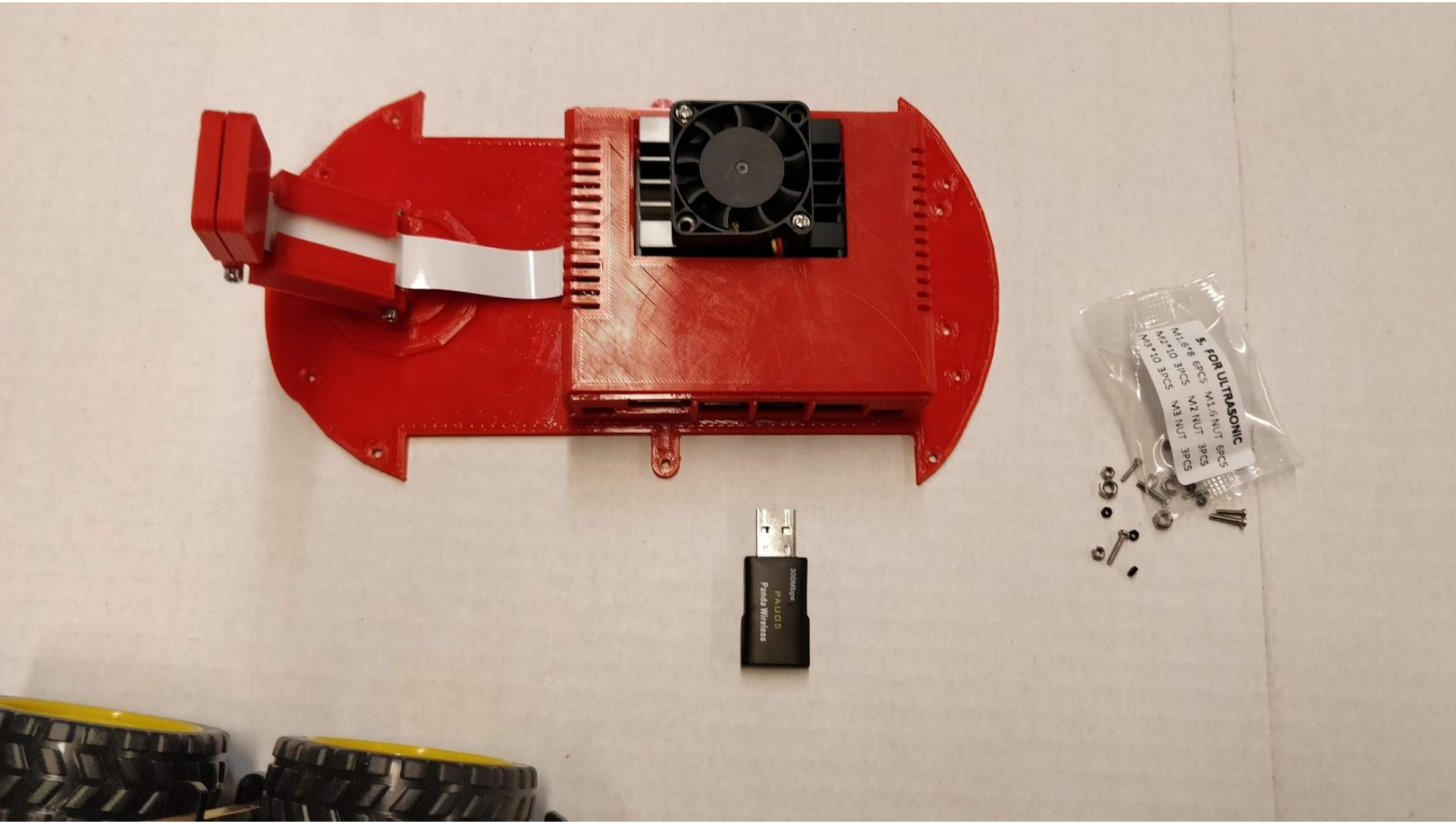
# Building the RC Car



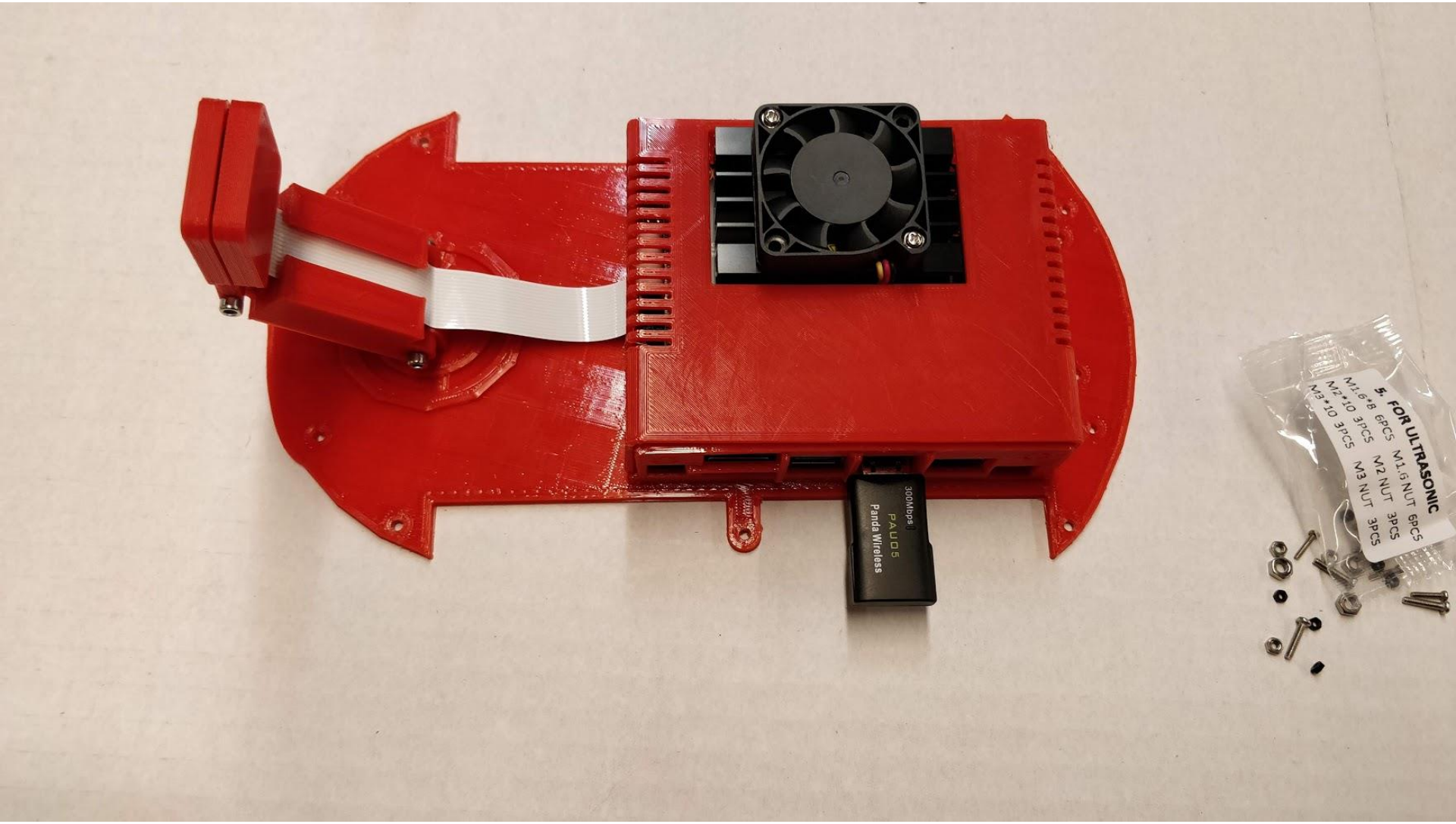
# Building the RC Car



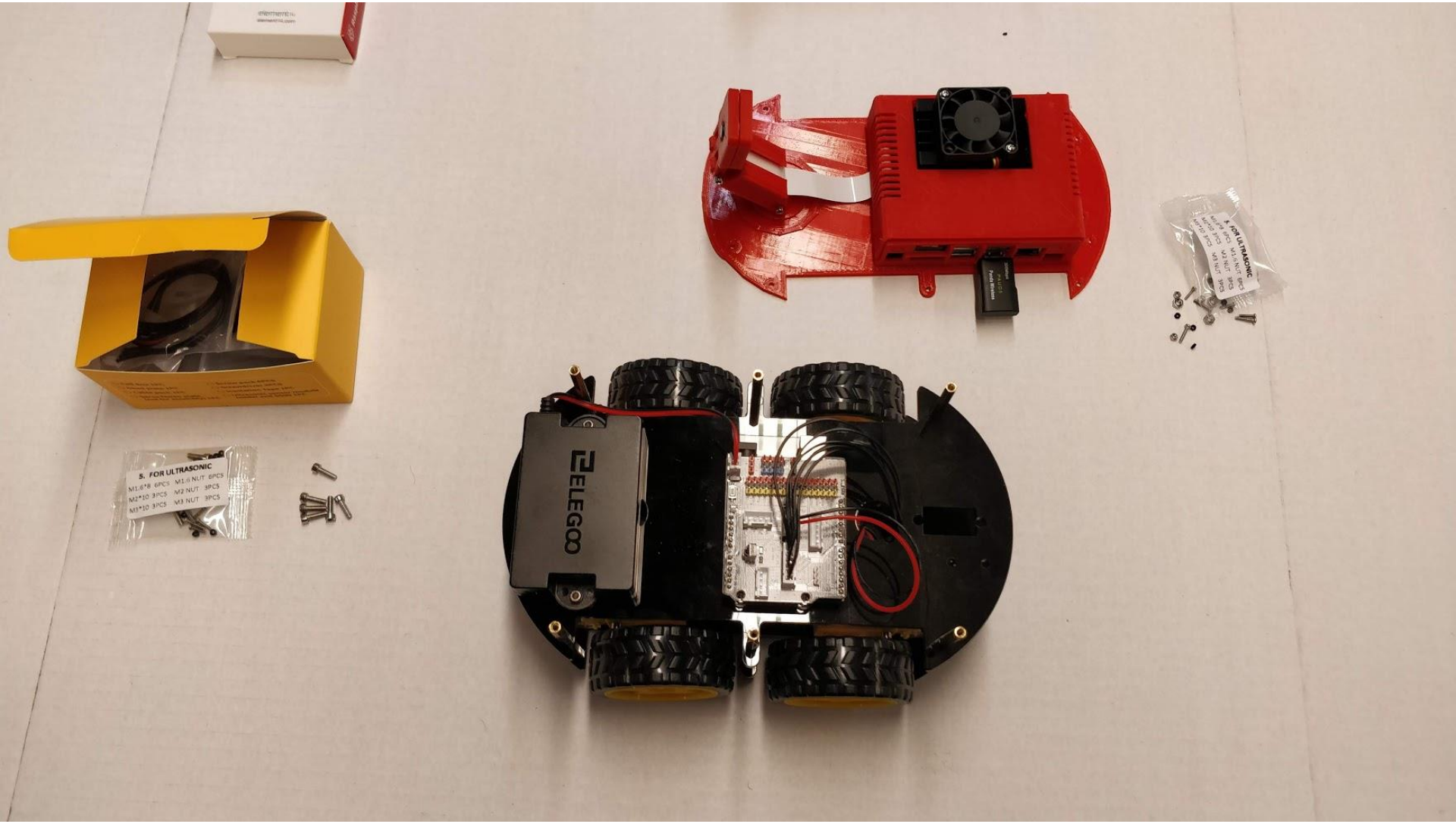
# Building the RC Car



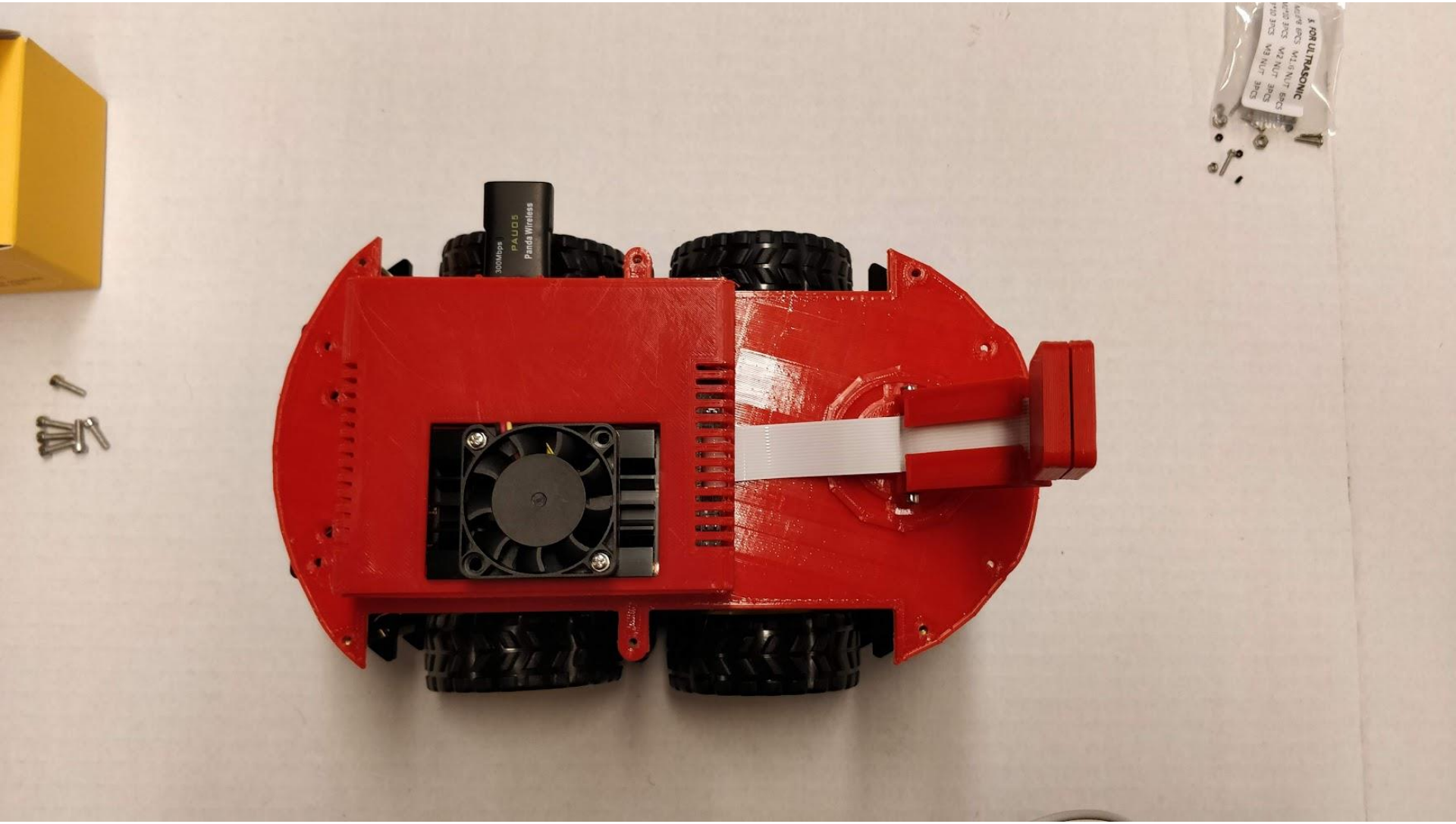
# Building the RC Car



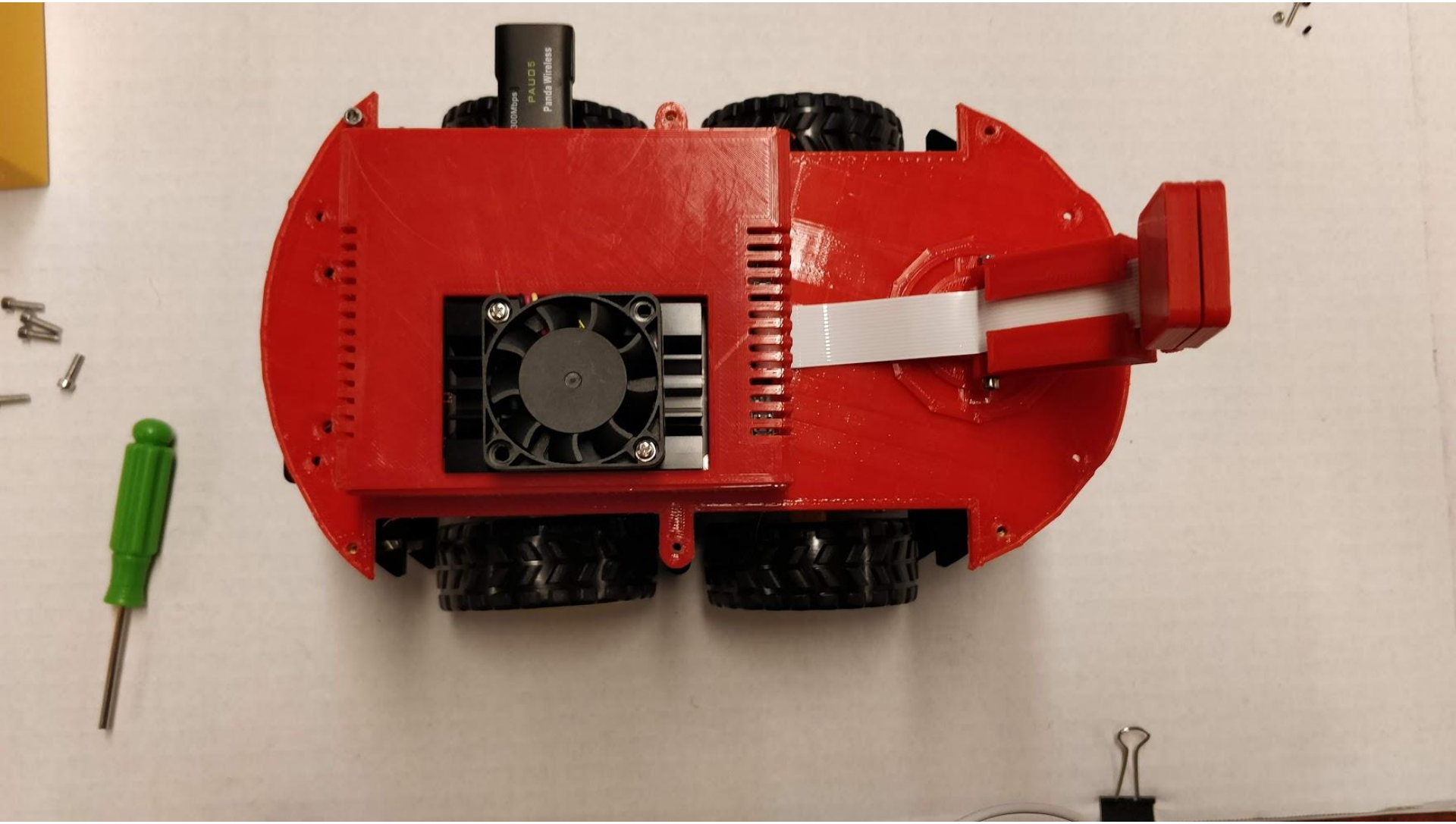
# Building the RC Car



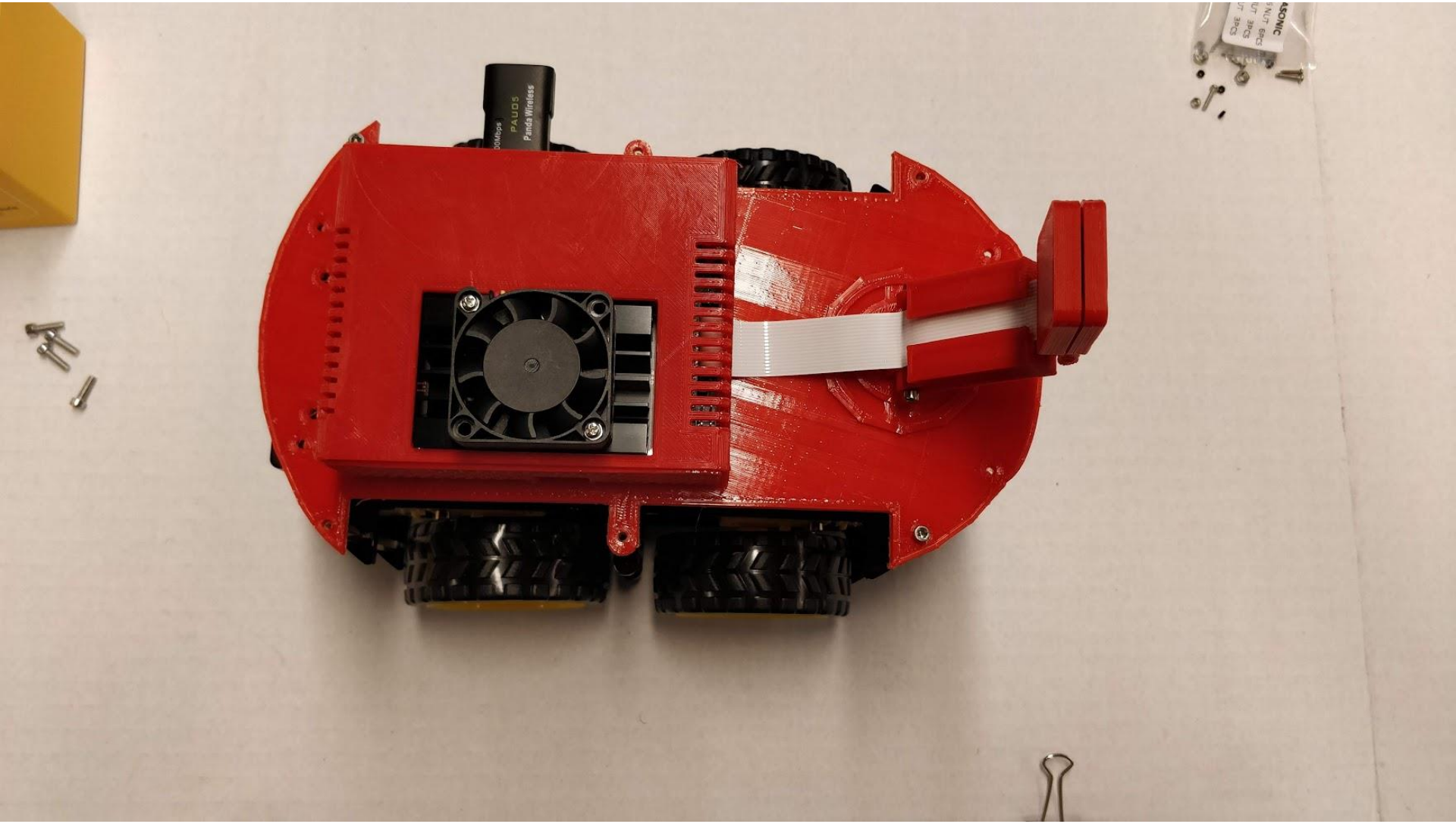
# Building the RC Car



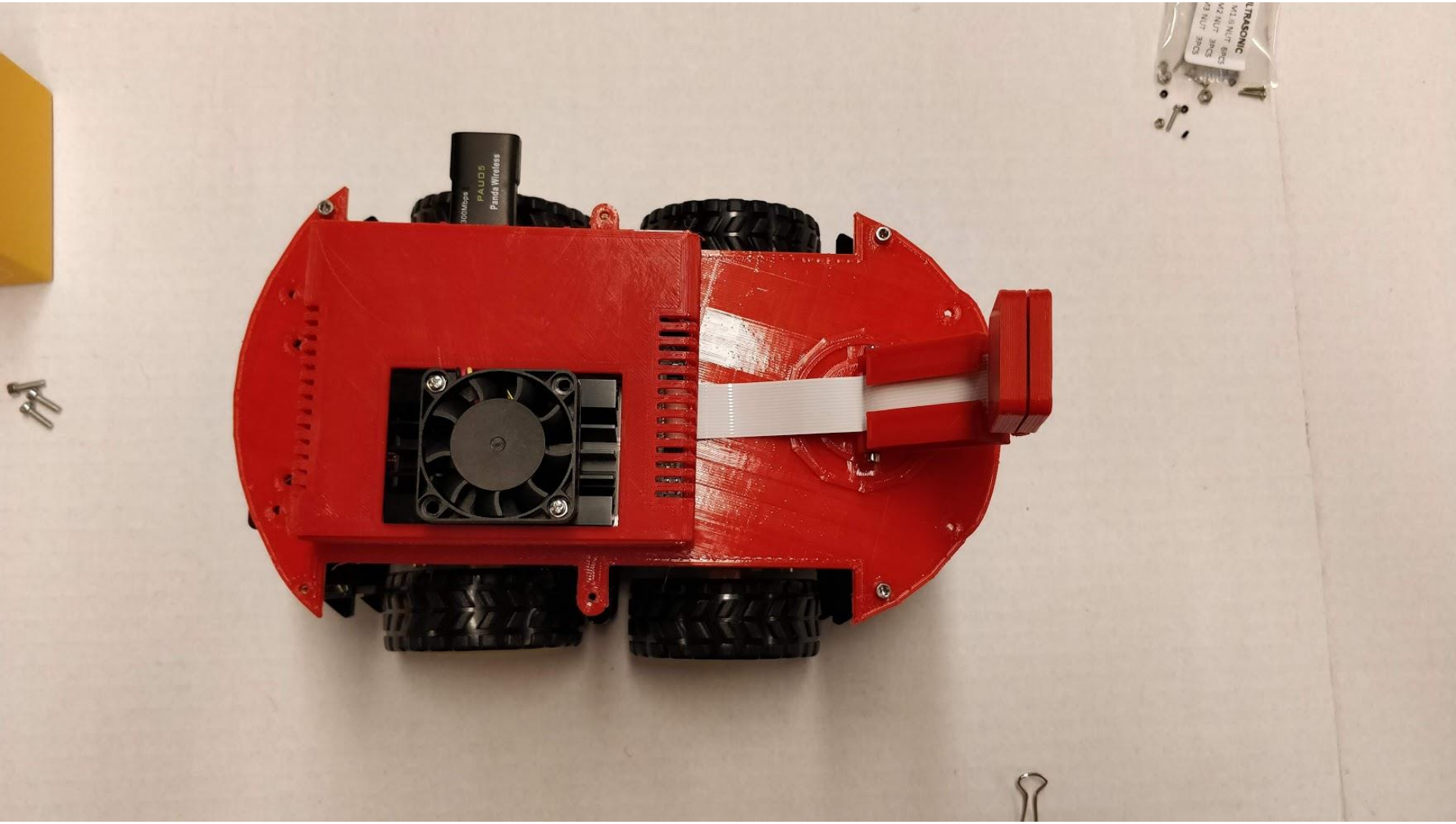
# Building the RC Car



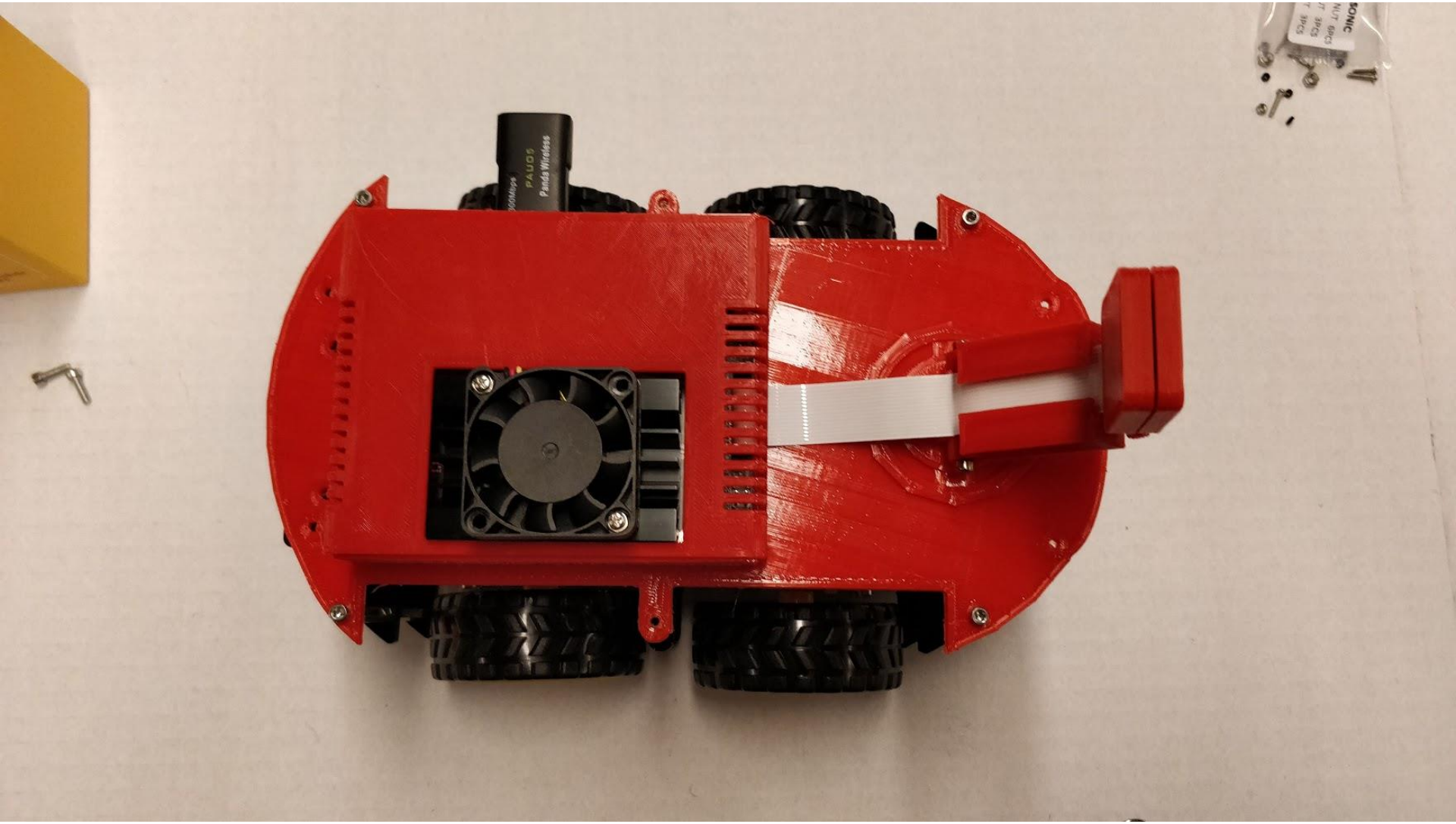
# Building the RC Car



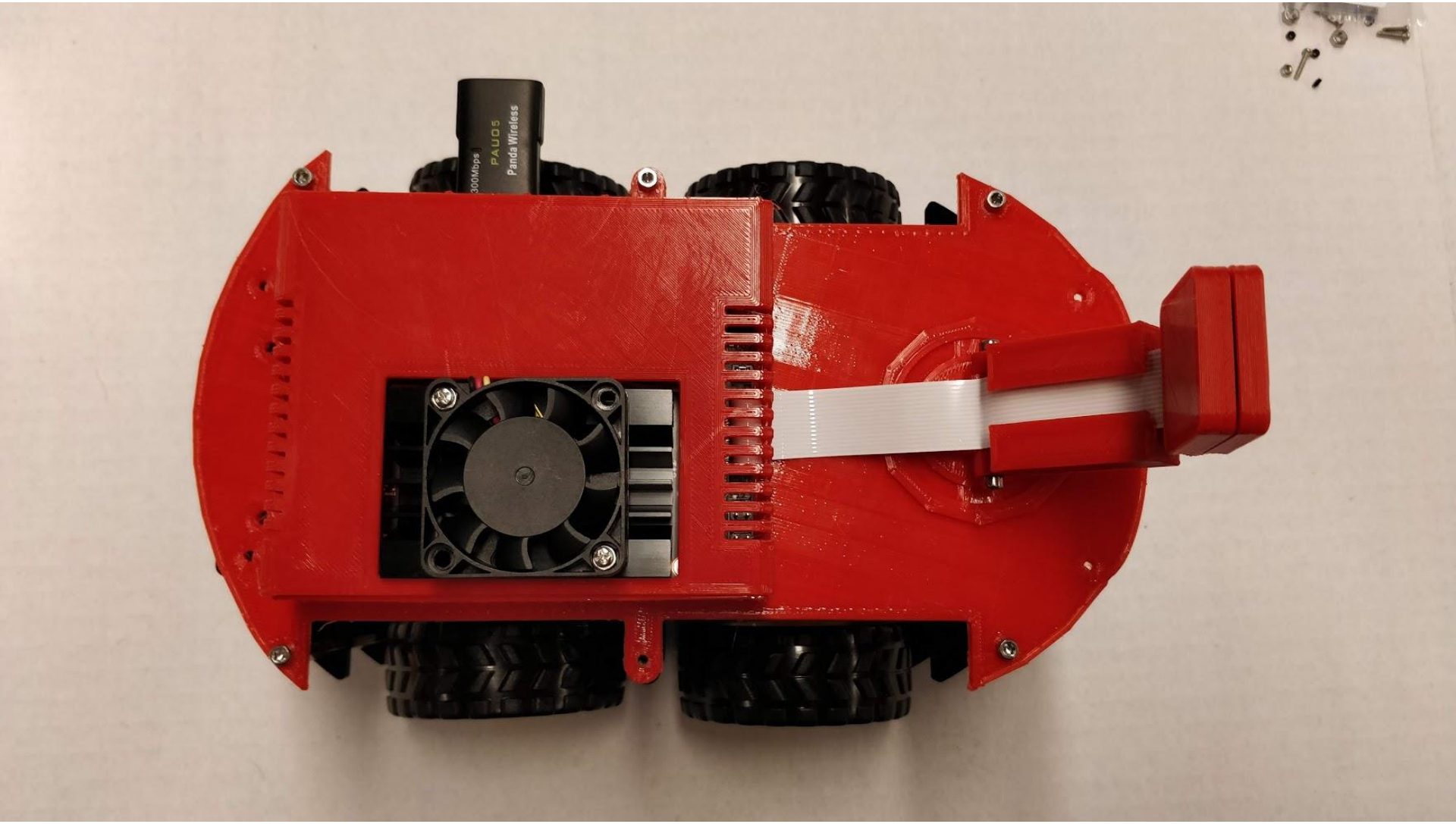
# Building the RC Car



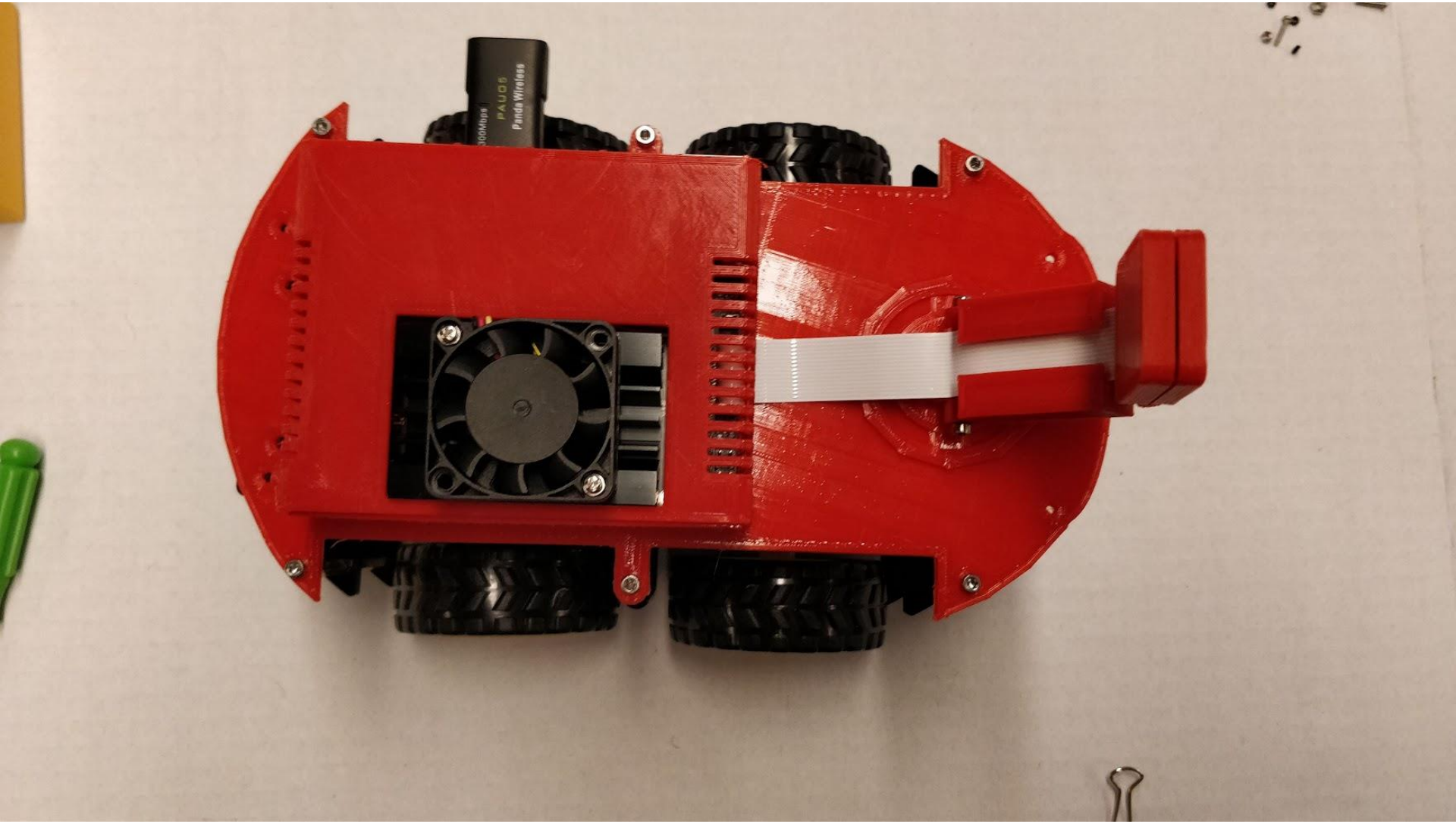
# Building the RC Car



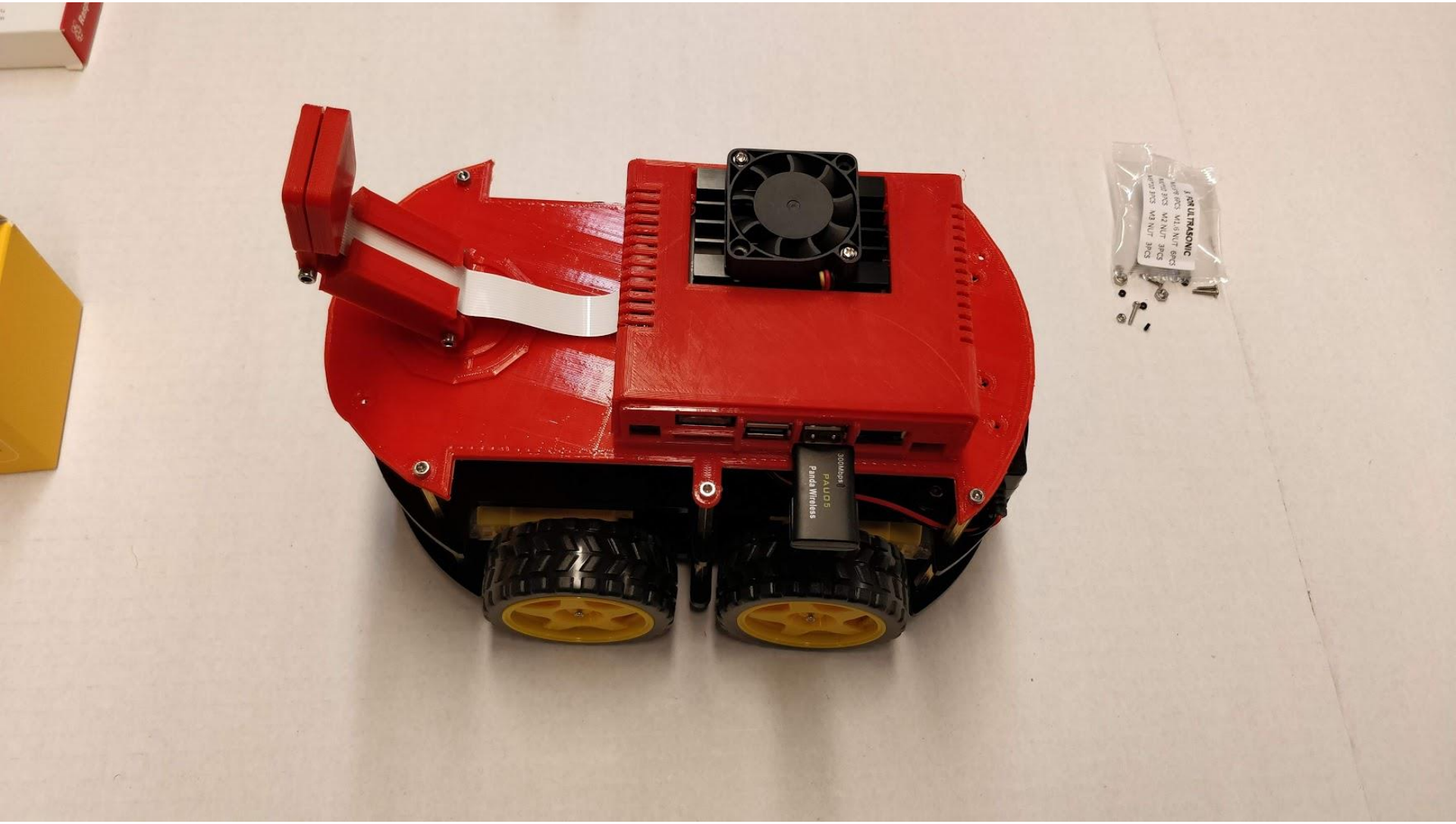
# Building the RC Car



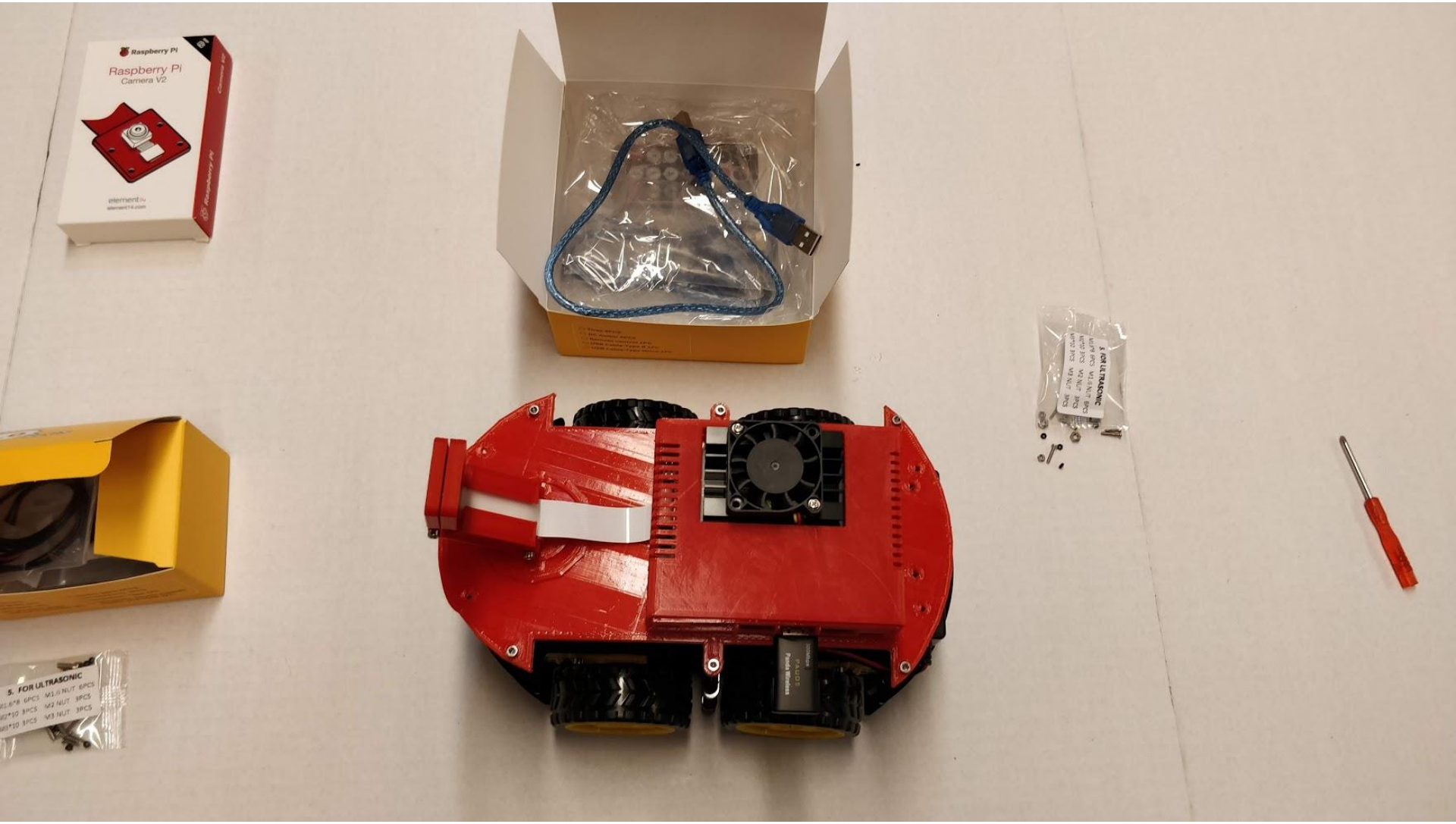
# Building the RC Car



# Building the RC Car



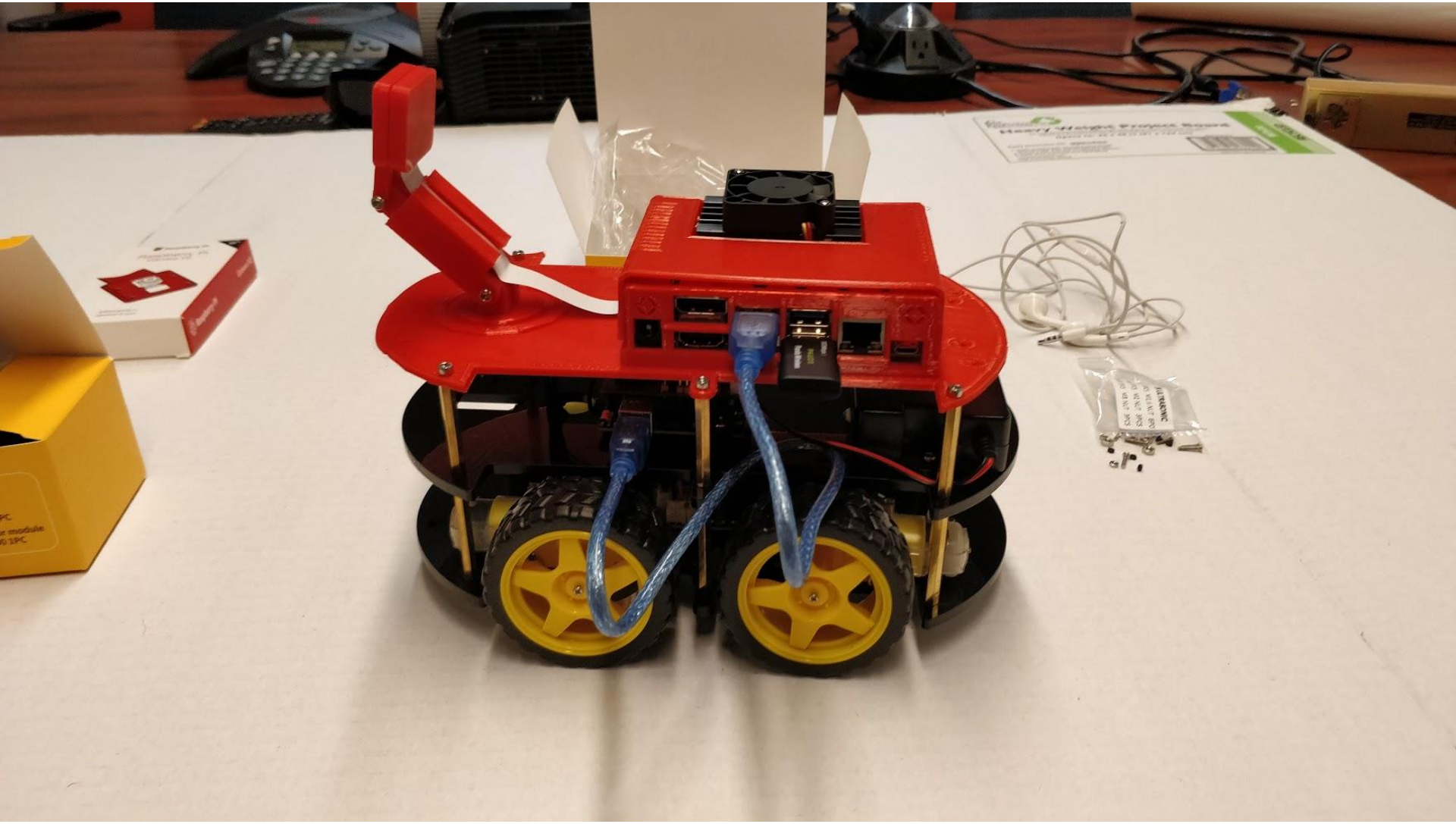
# Building the RC Car



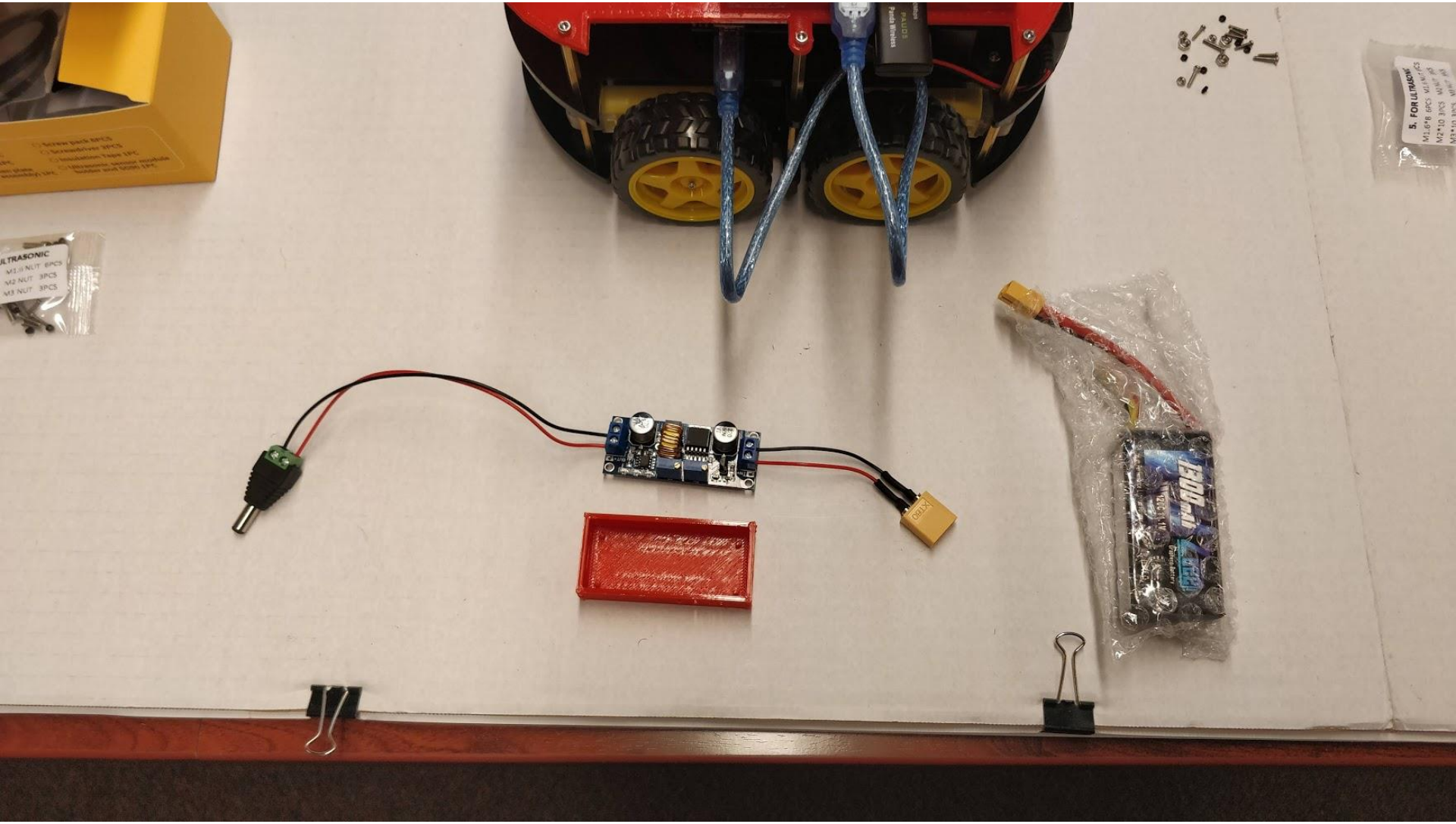
Joint Institute for Computational Sciences

# Building the RC Car

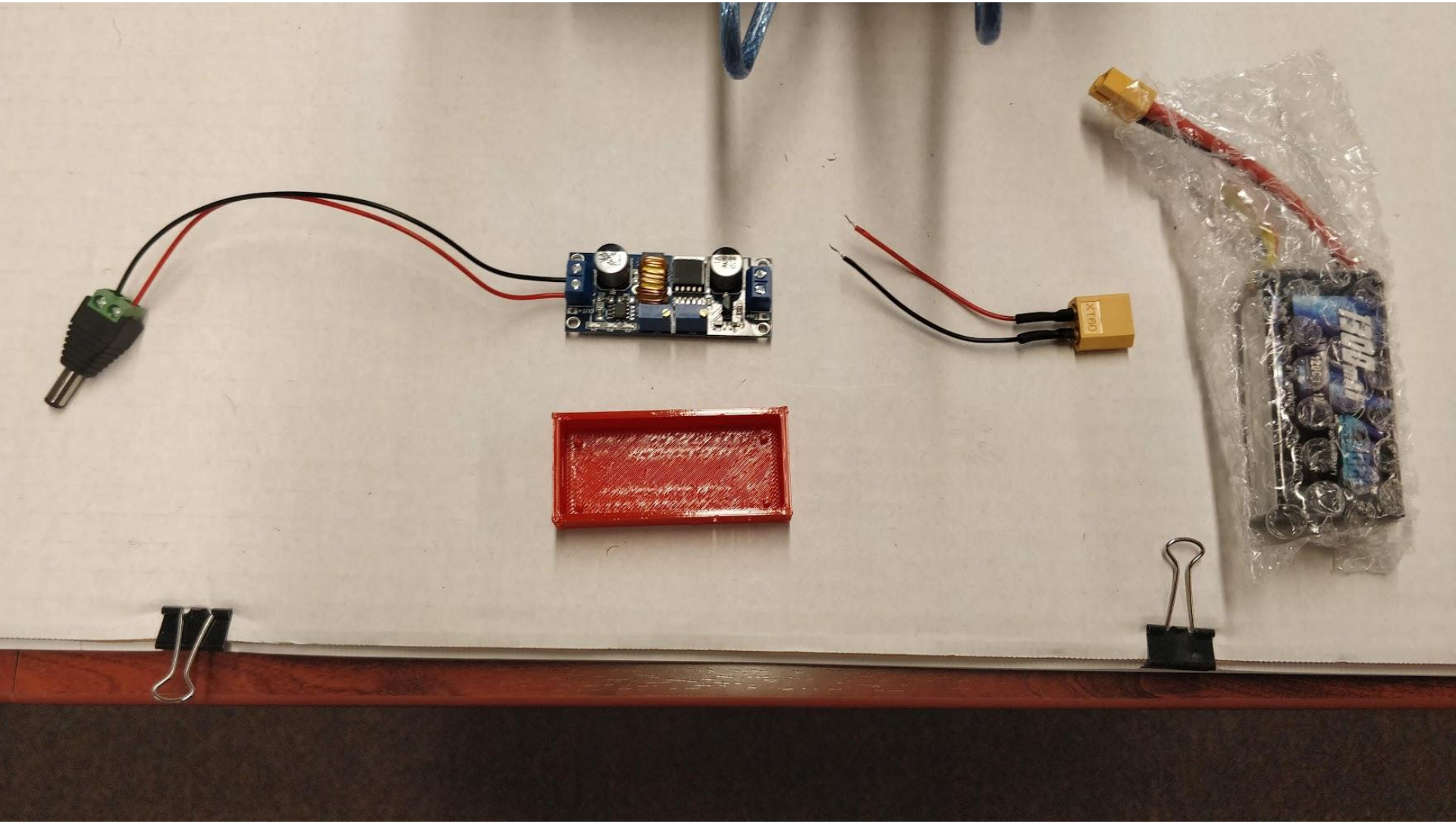
INNOVATIVE COMPUTING LABORATORY



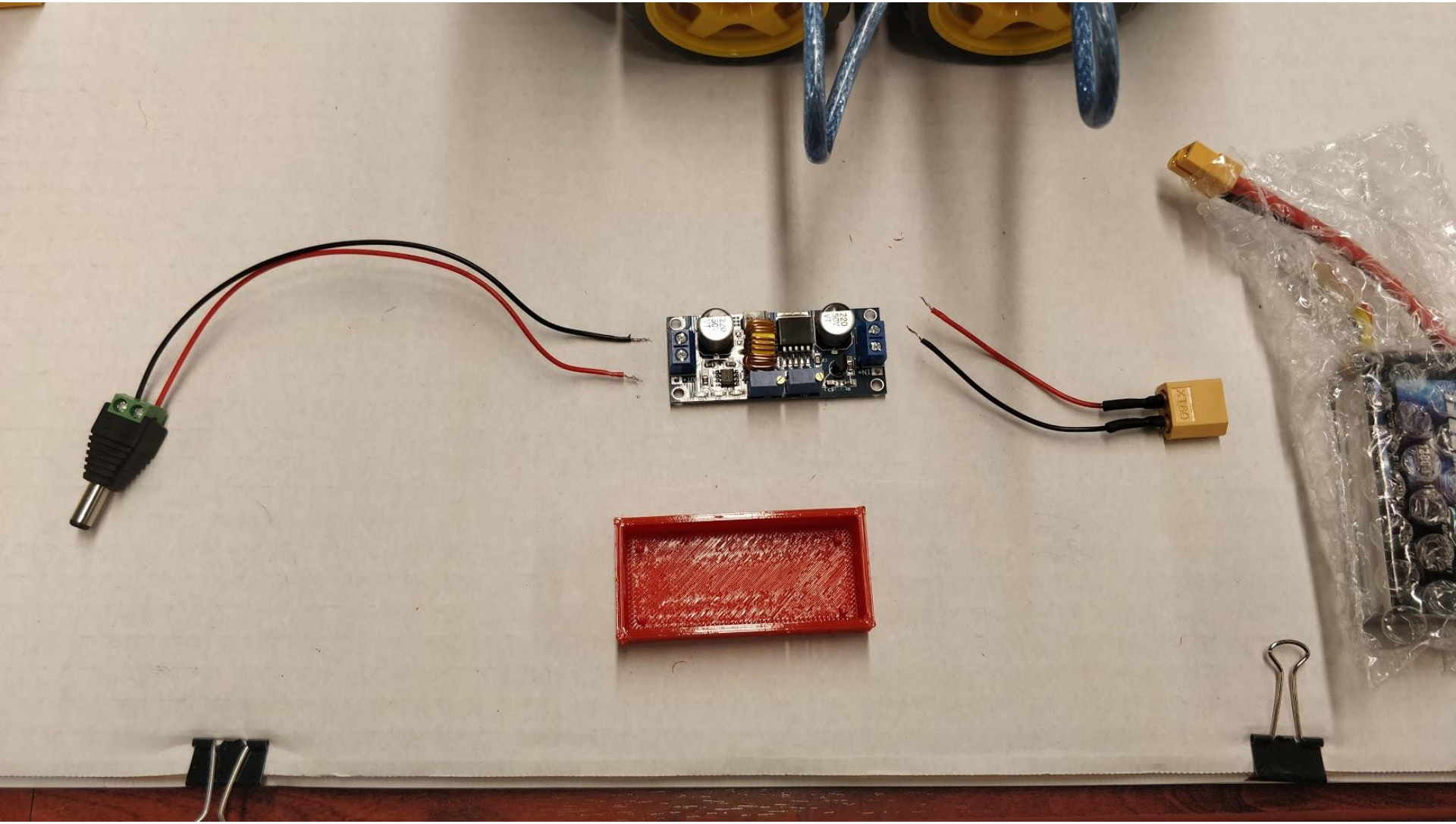
# Building the RC Car

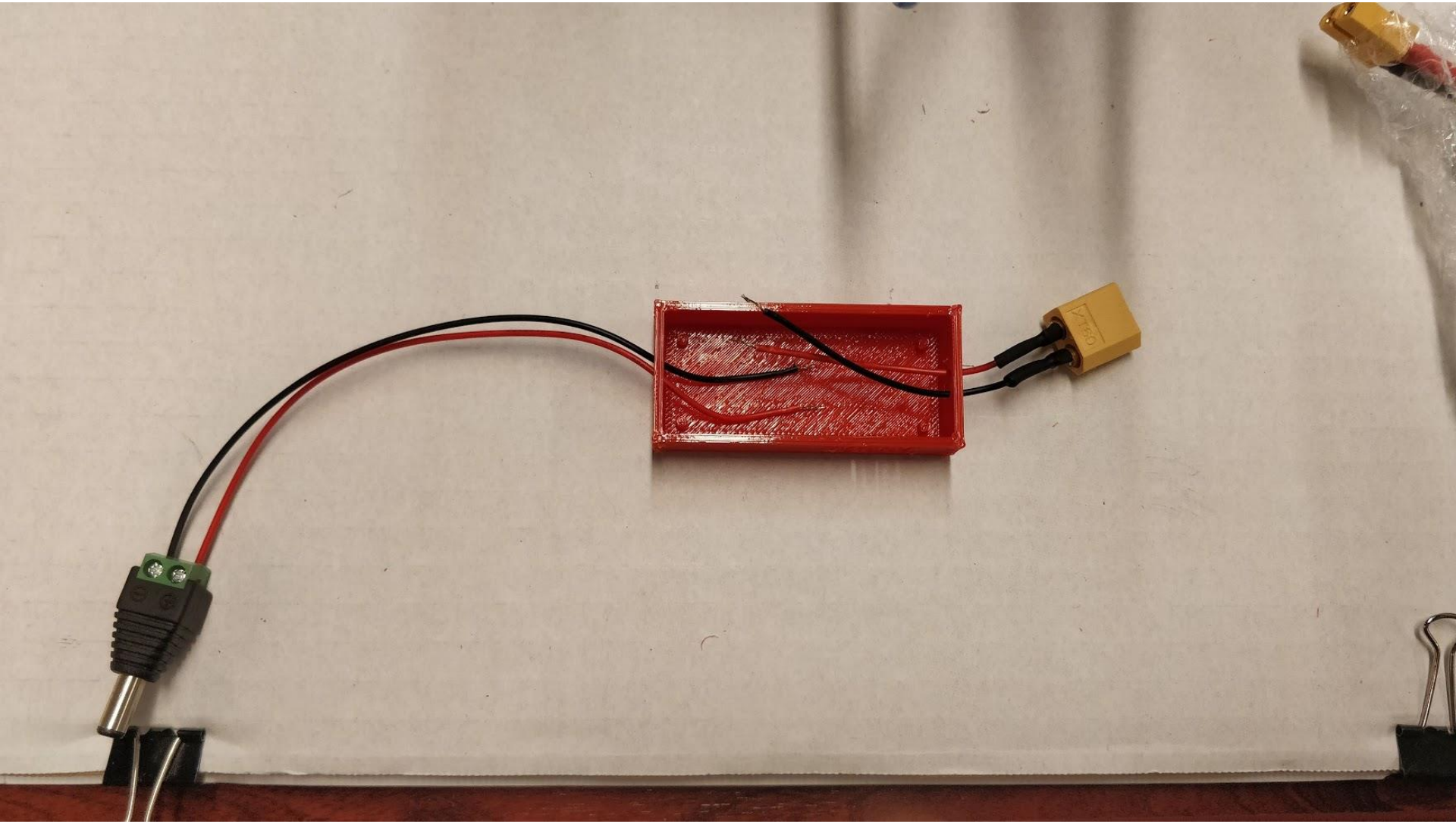


# Building the RC Car

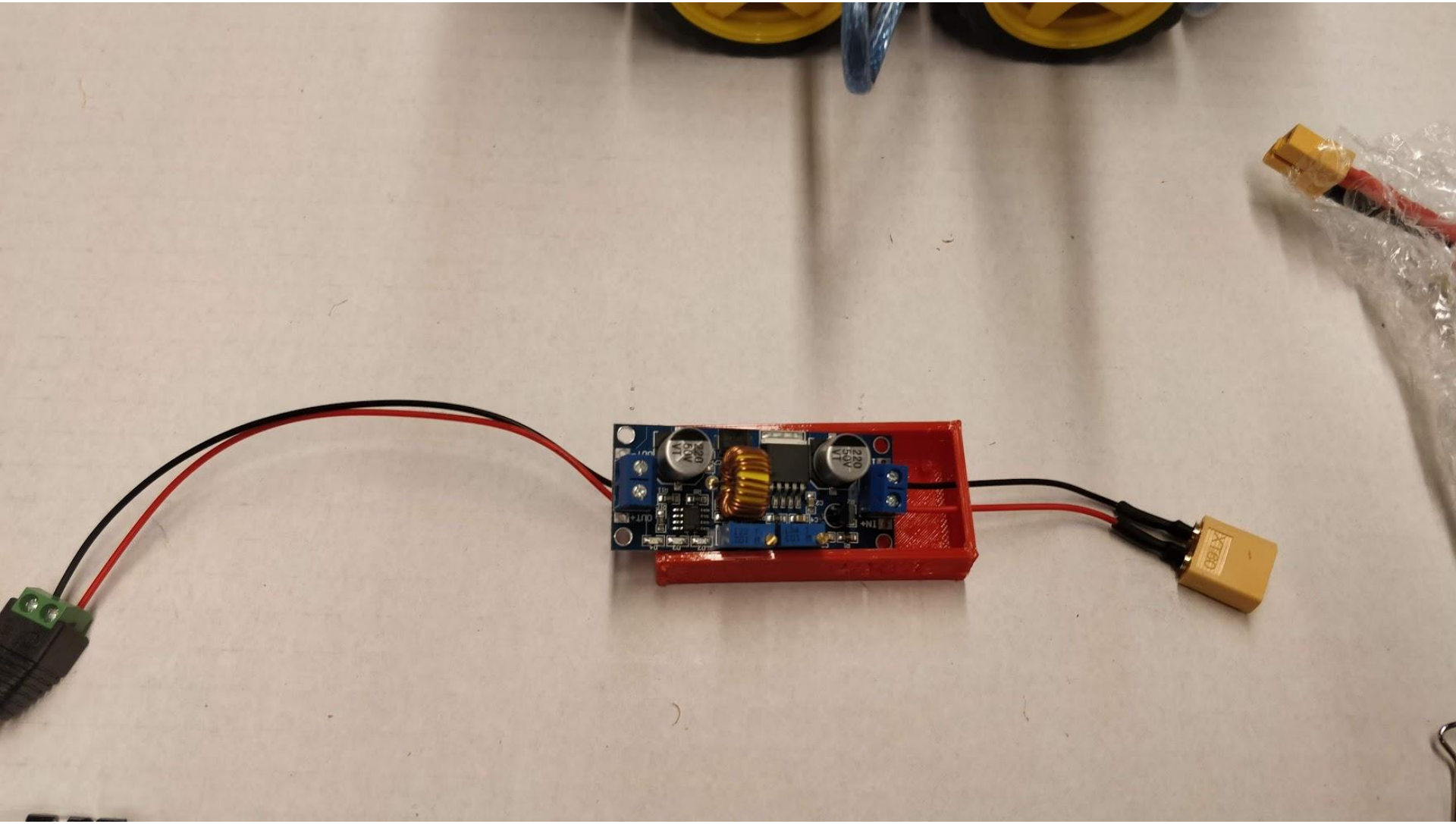


# Building the RC Car



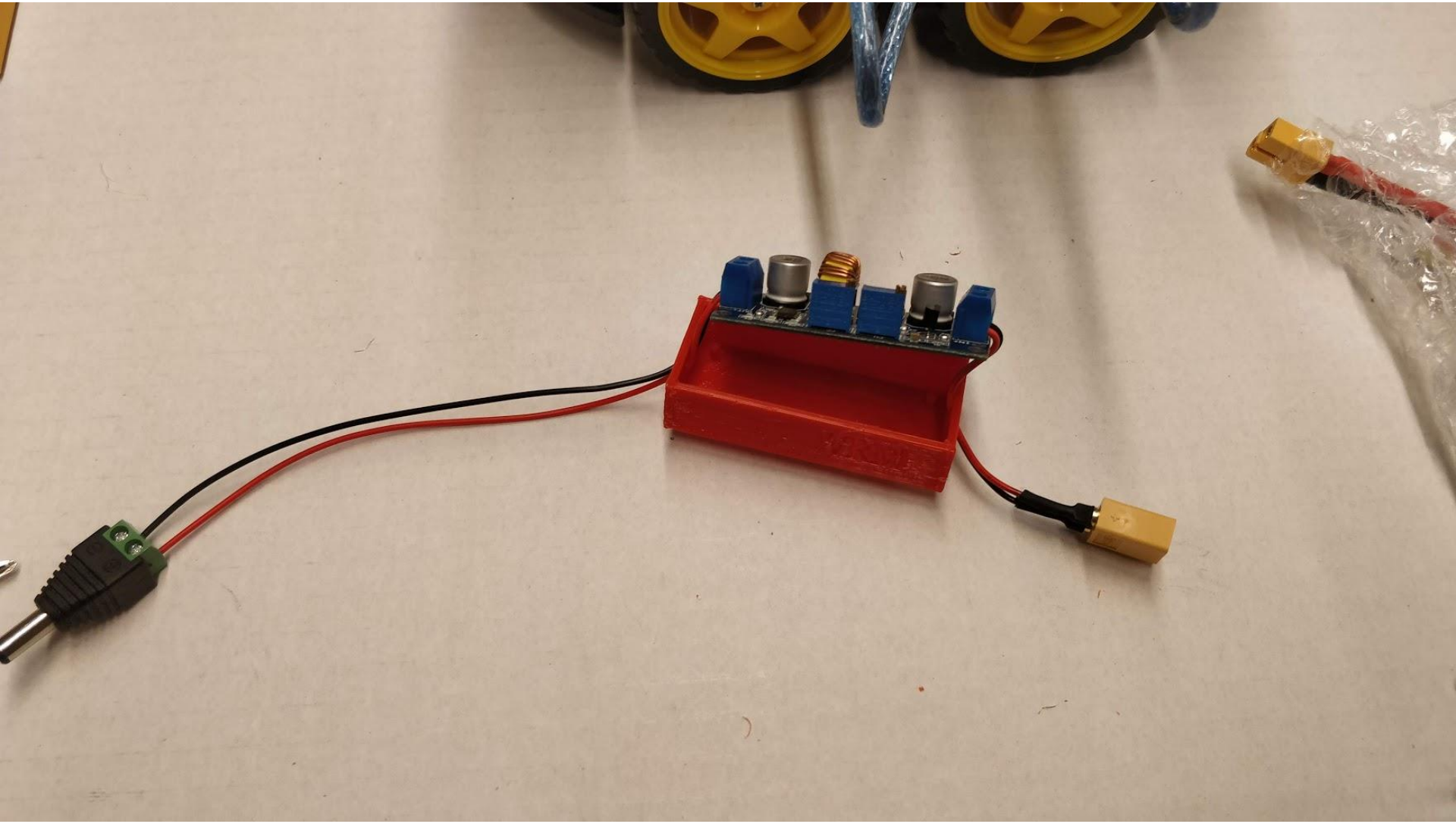


# Building the RC Car

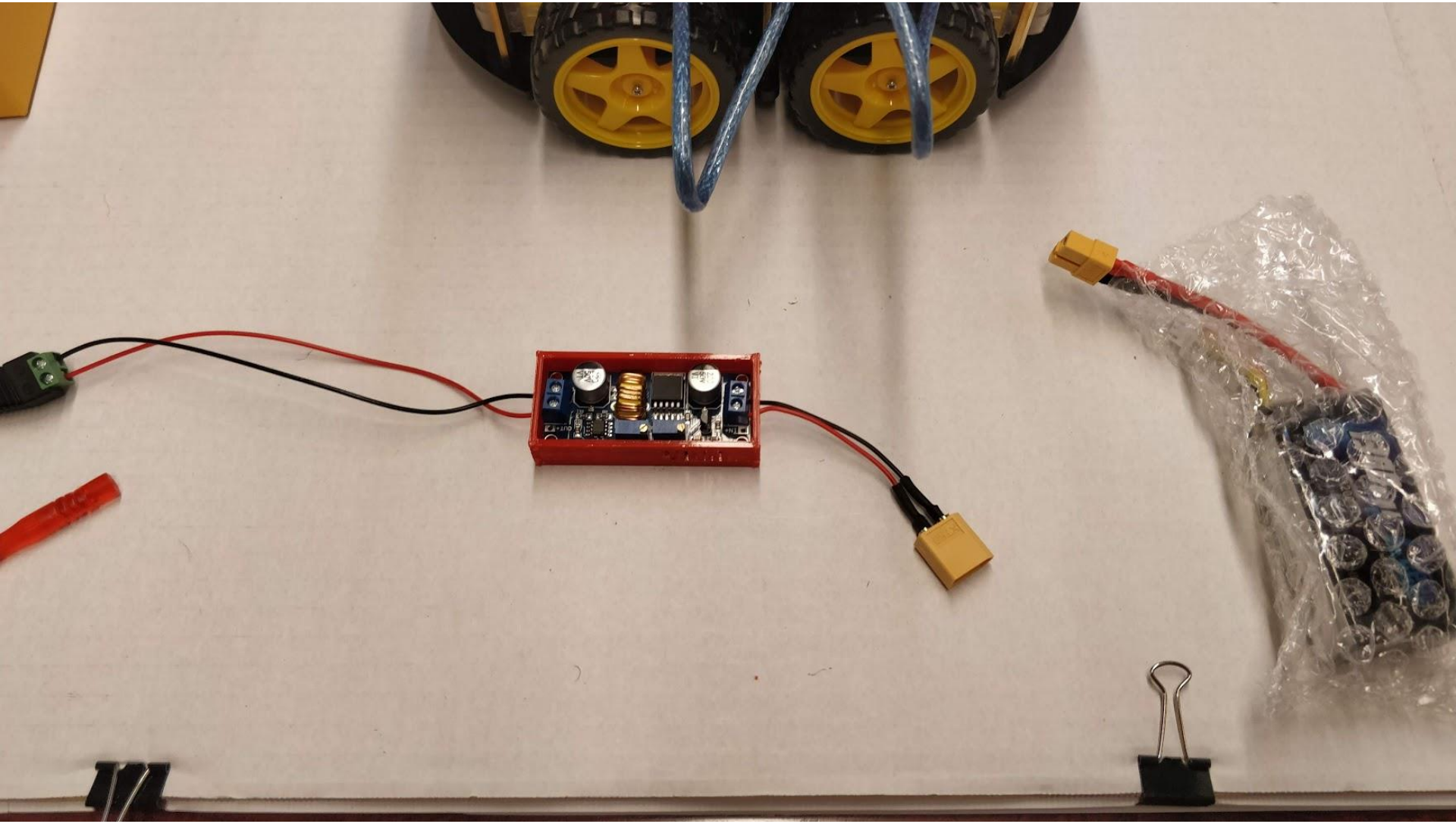


# Building the RC Car

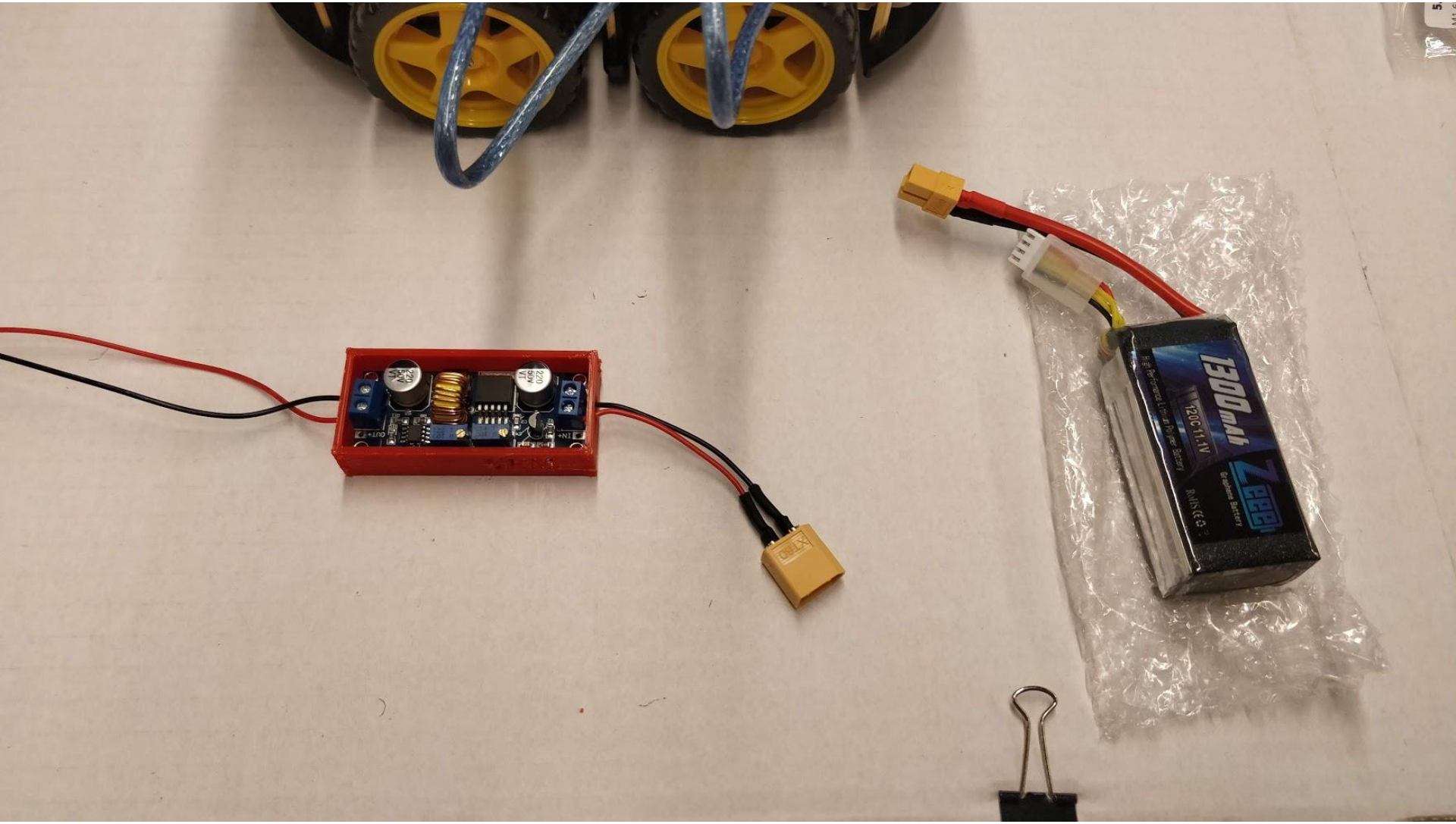




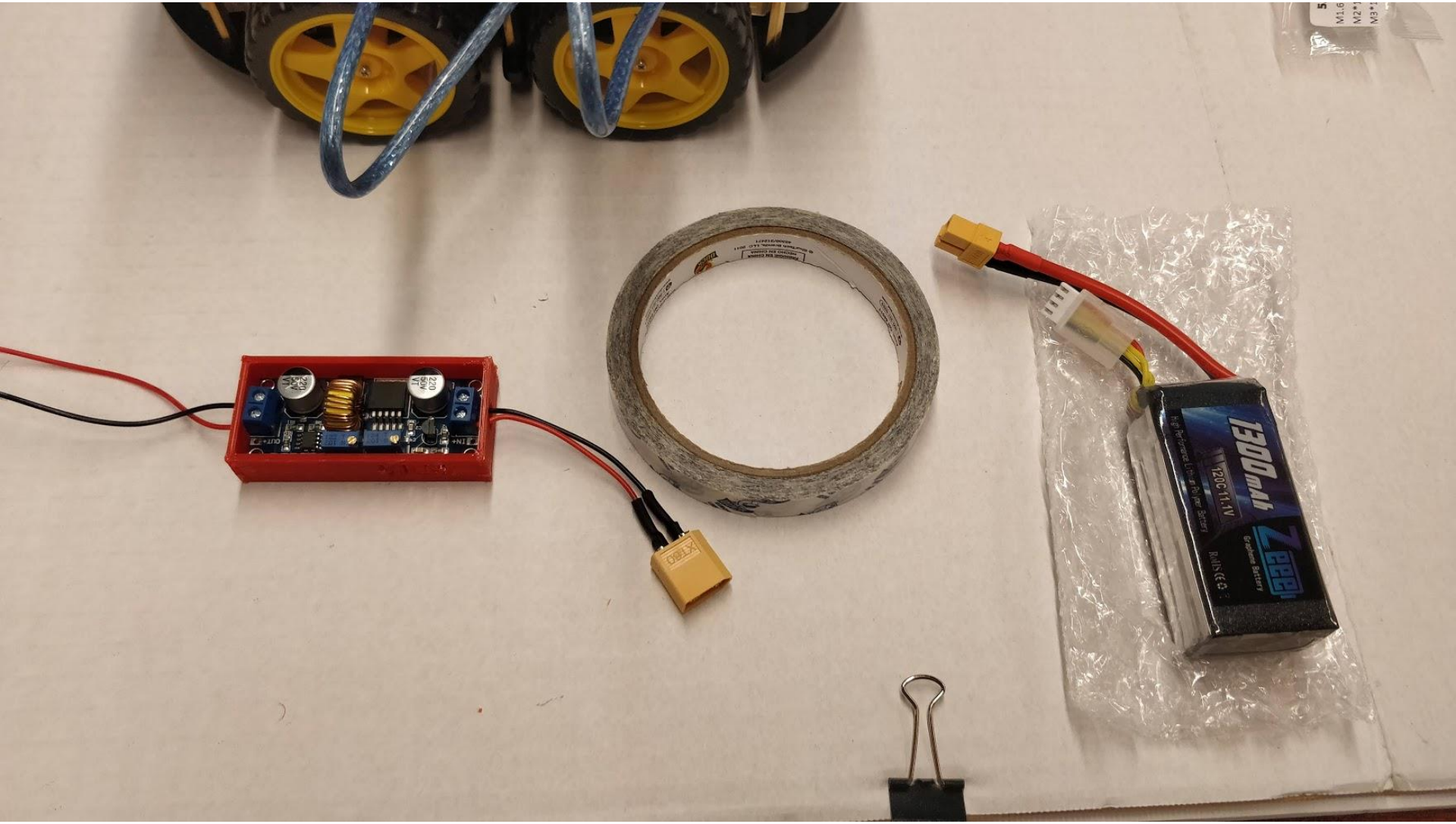
# Building the RC Car



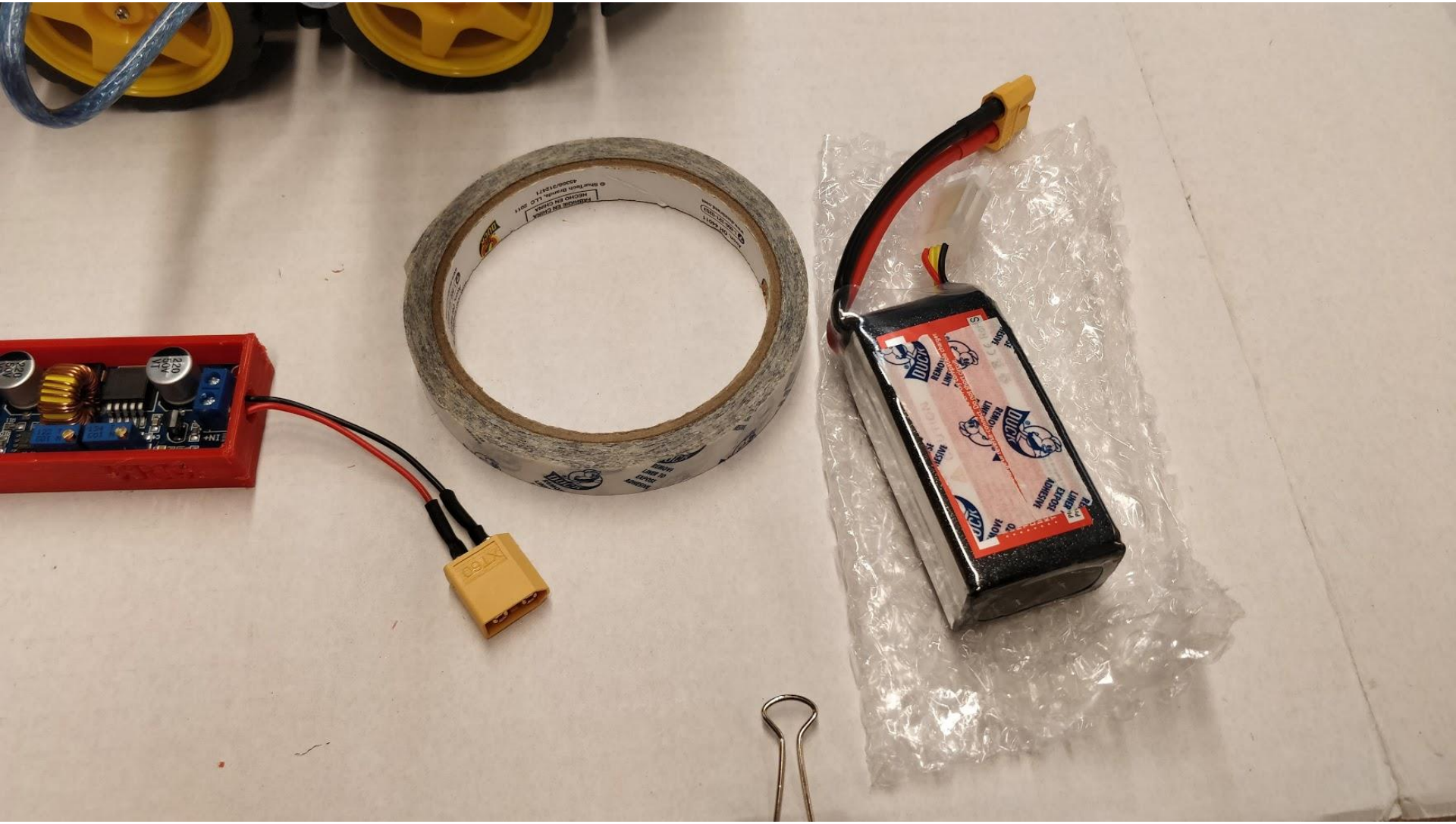
# Building the RC Car



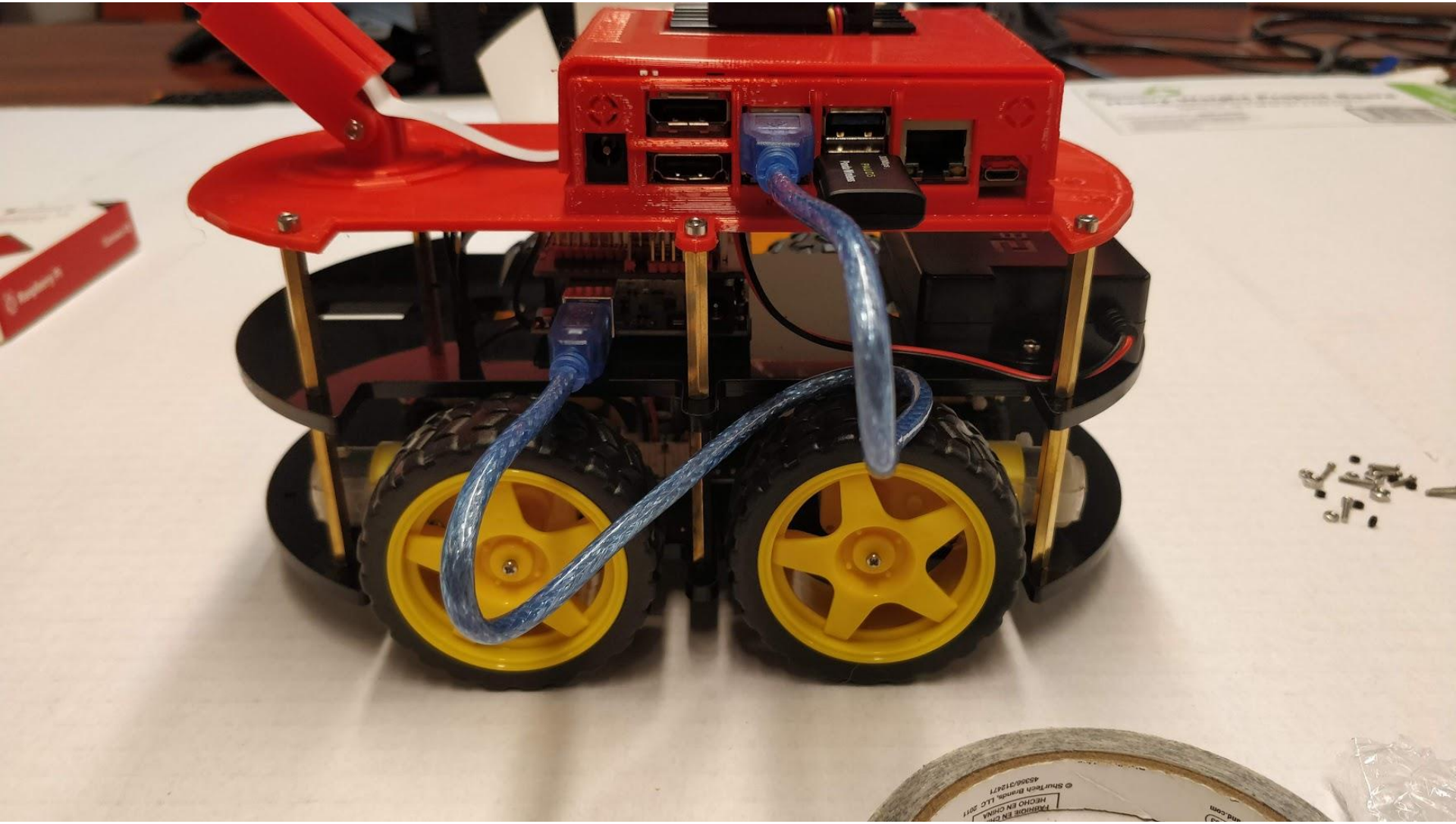
# Building the RC Car



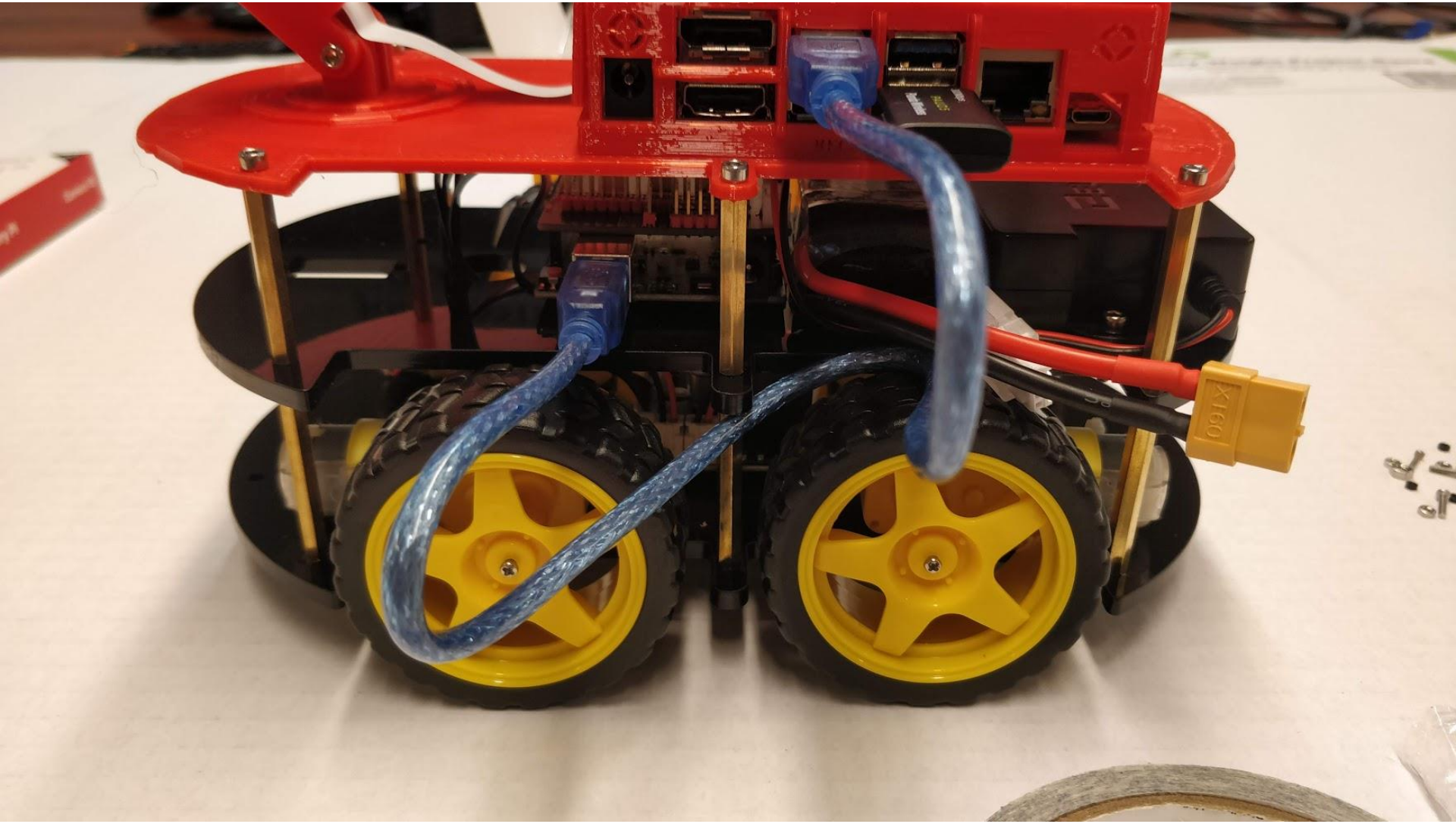
# Building the RC Car



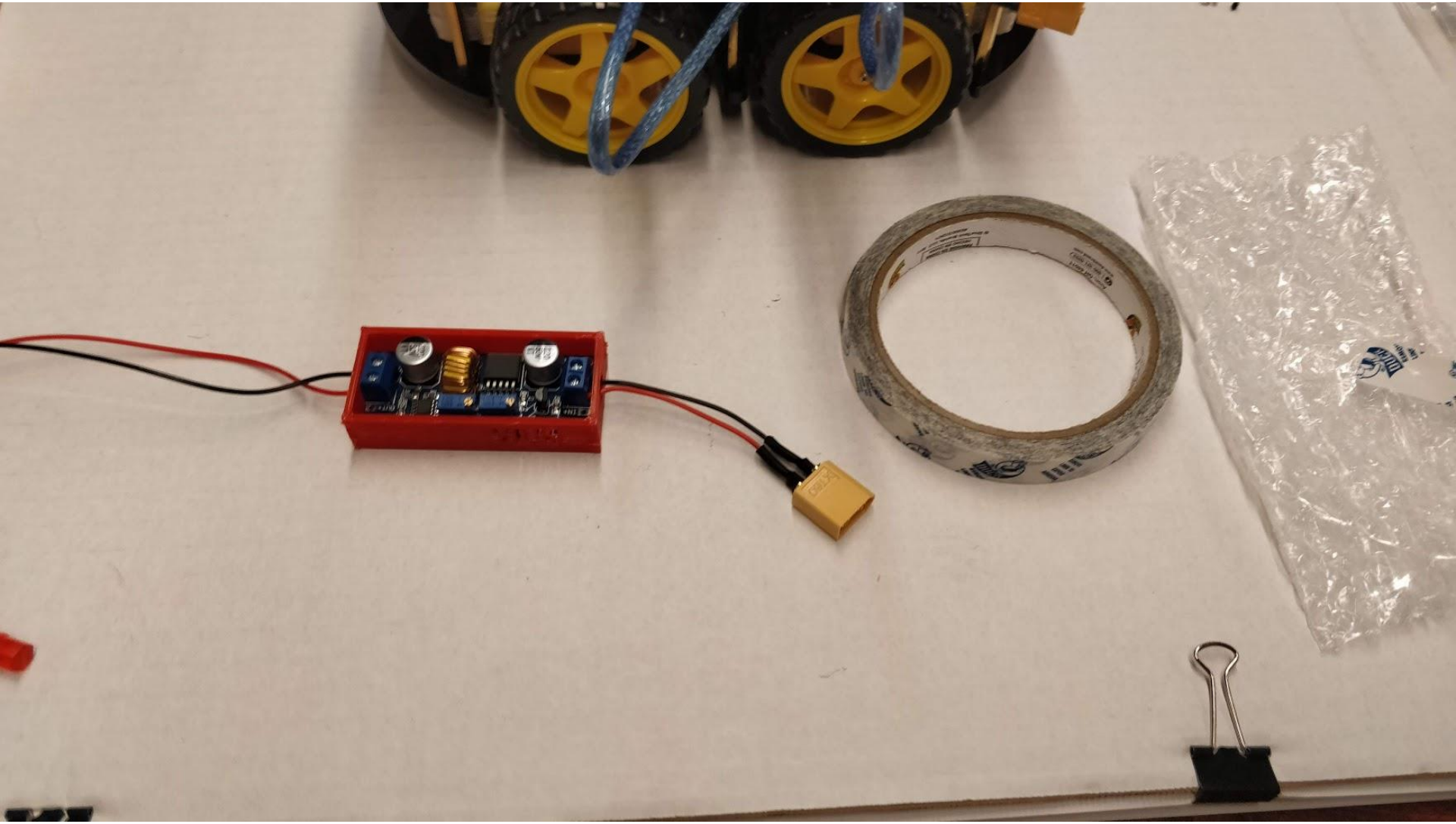
# Building the RC Car

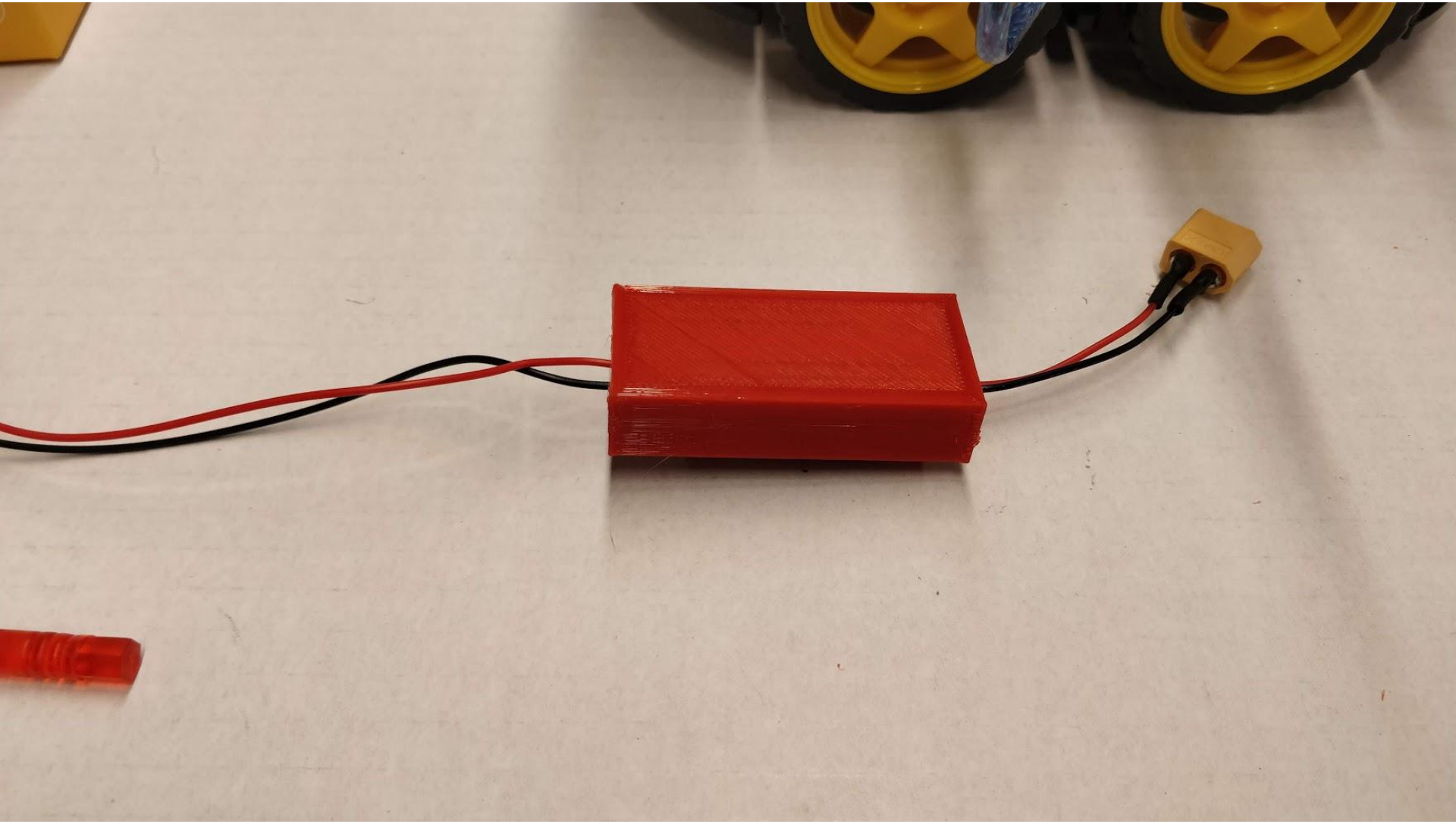


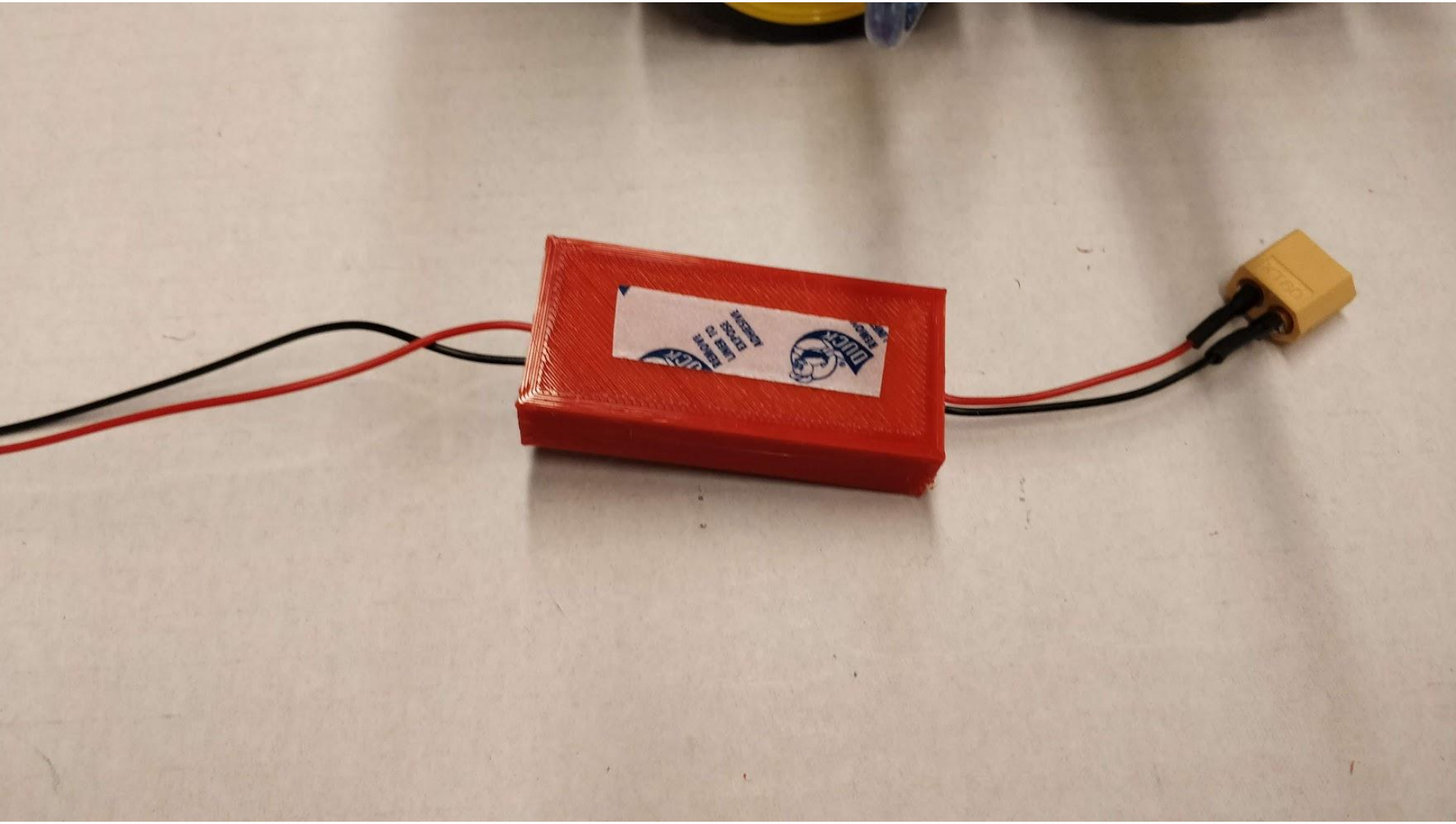
# Building the RC Car



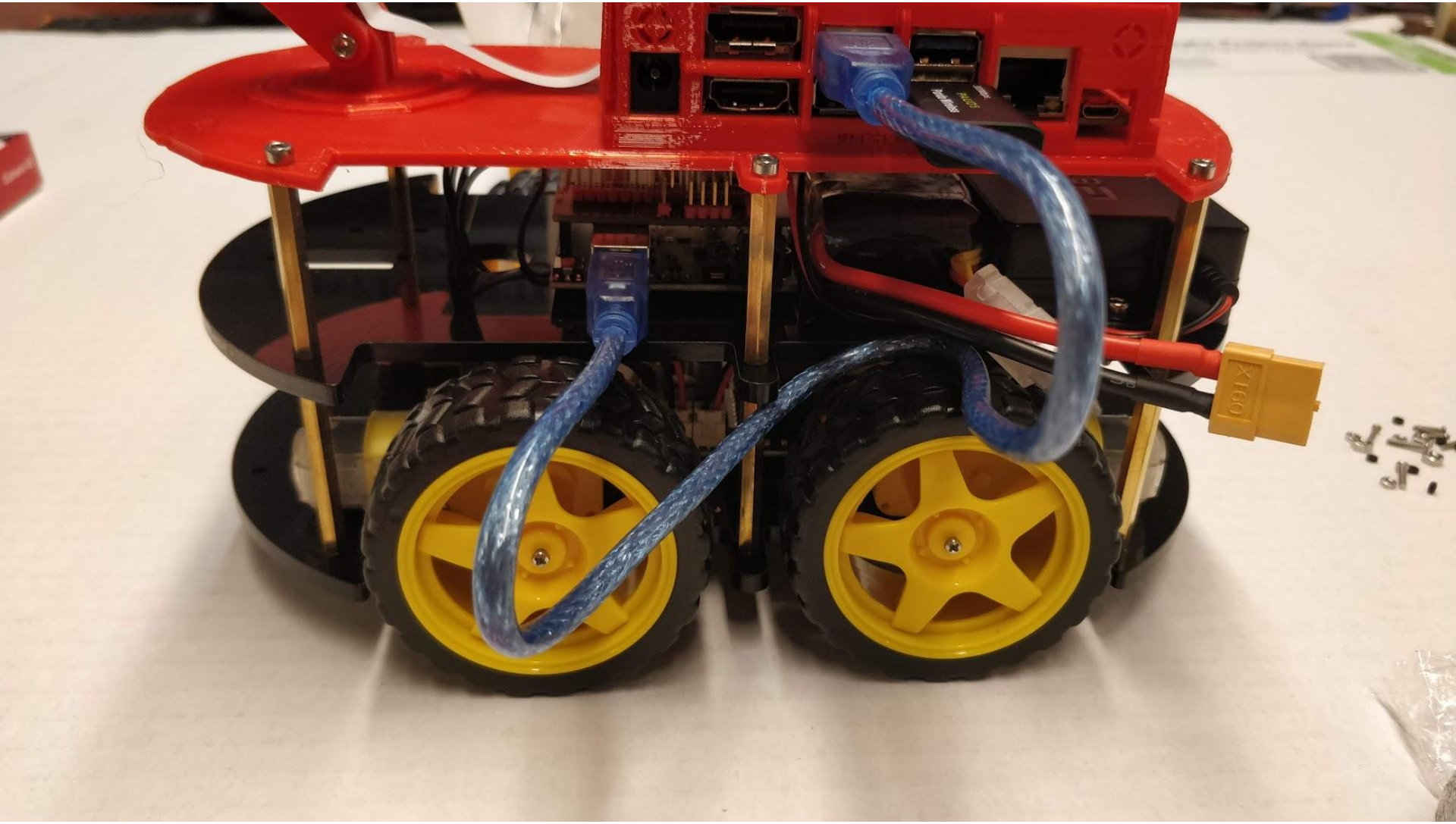
# Building the RC Car



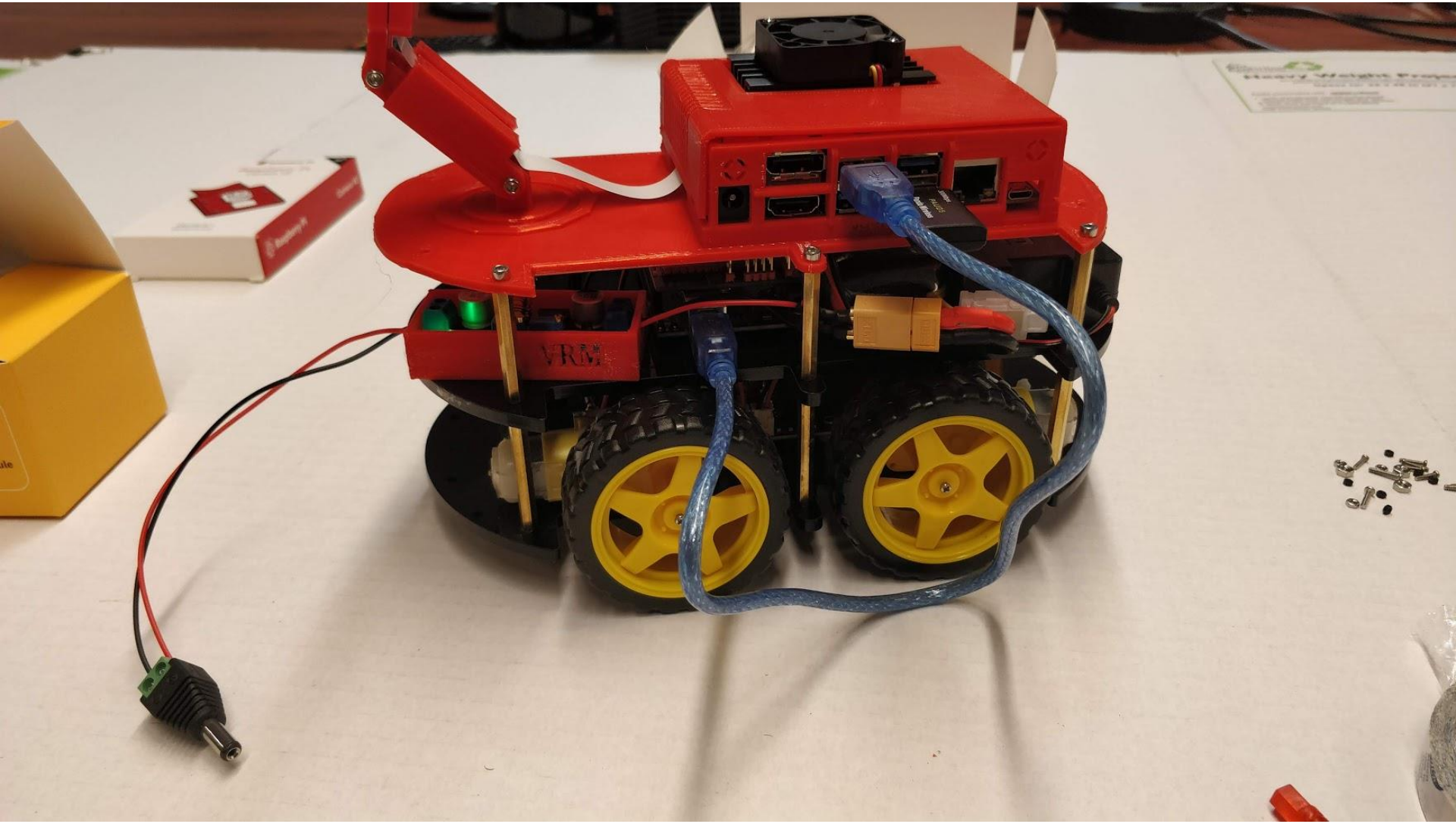




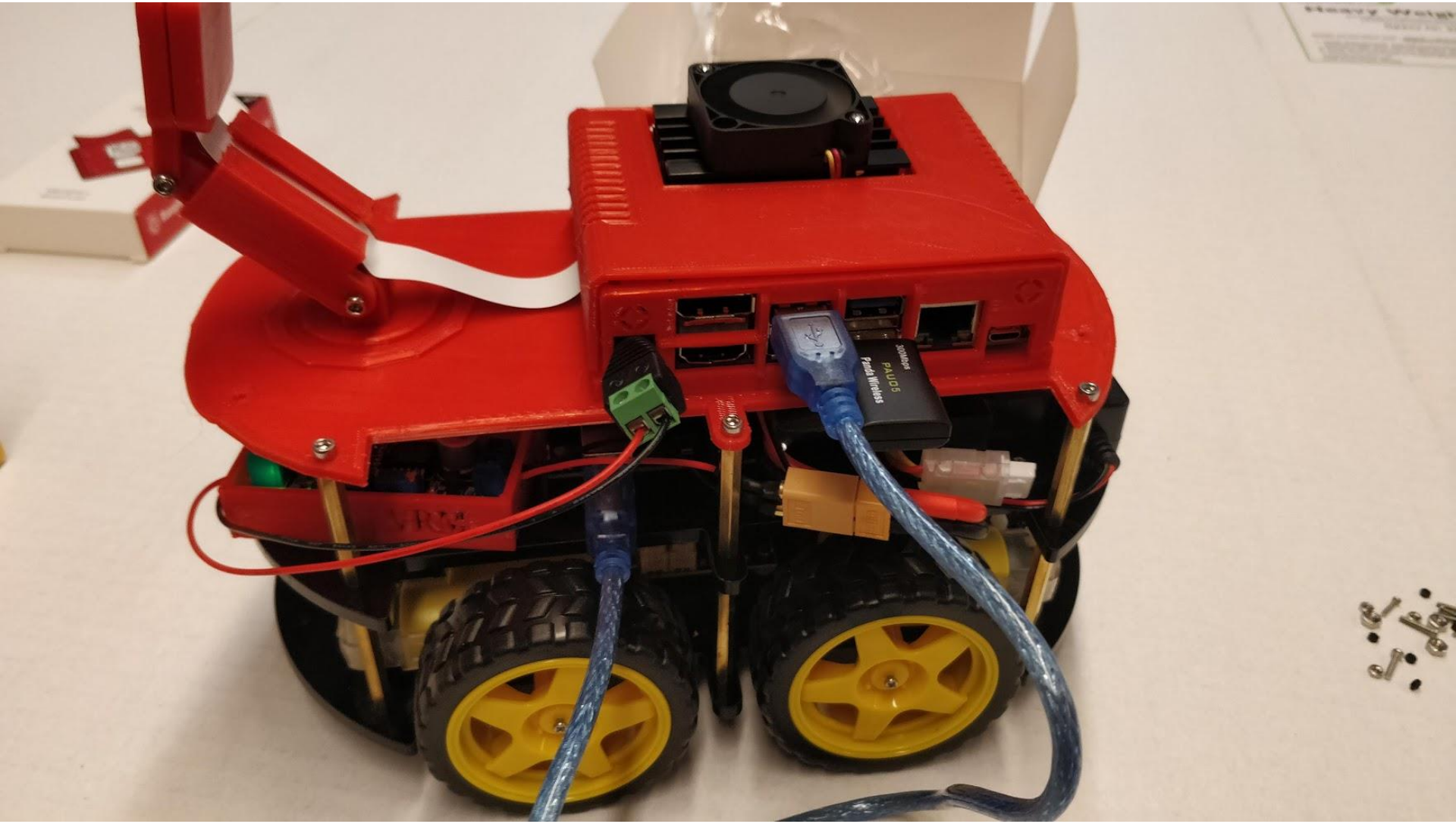
# Building the RC Car



# Building the RC Car



# Building the RC Car



# Software Setup

Download the Jetson Nano Developer Kit SD Card Image, and note where it was saved on the computer. https://developer.nvidia.com/jetson-nano-sd-card-image

Use Etcher to write the Jetson Nano Developer Kit SD Card Image to your microSD card

Download, install, and launch Etcher. https://www.balena.io/etcher

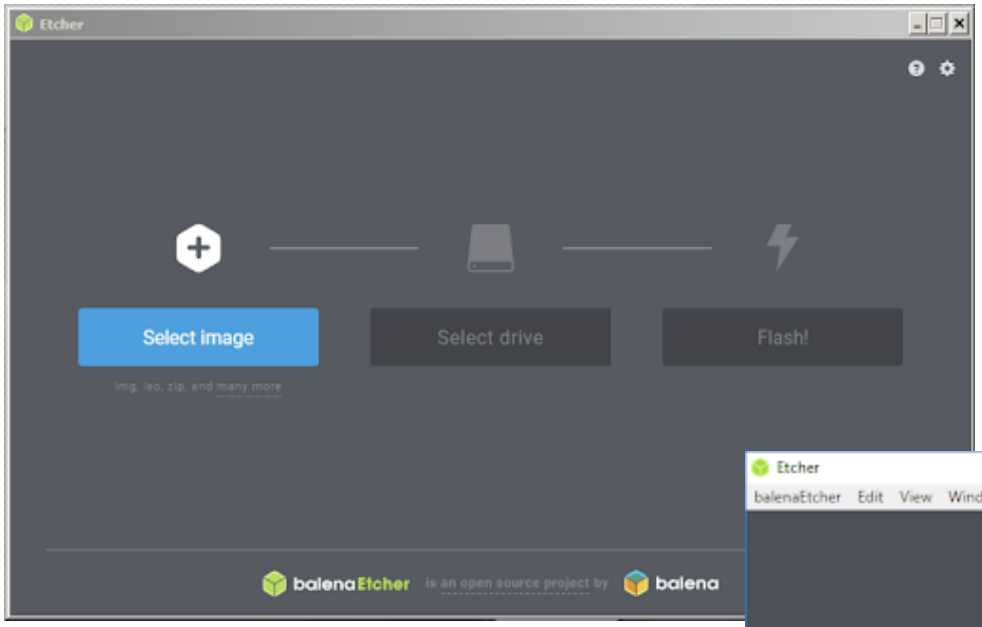
Click “Select image” and choose the zipped image file downloaded earlier.
Insert your microSD card if not already inserted.
Click Cancel (per this explanation) if Windows prompts you with a dialog like this:

Click “Select drive” and choose the correct device.

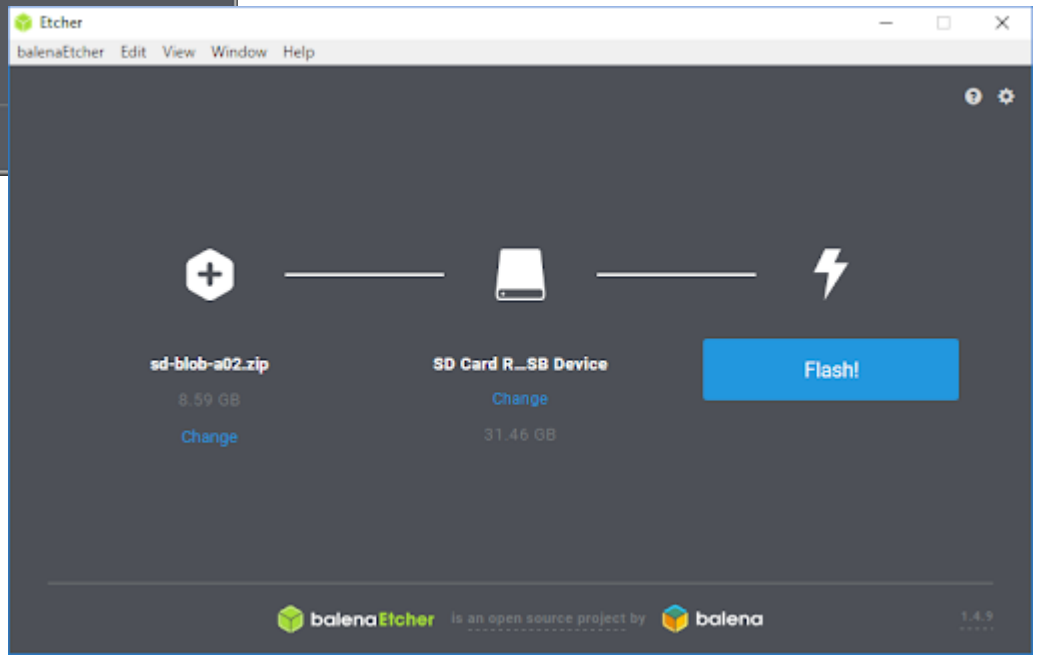
Click “Flash!” It will take Etcher about 10 minutes to write and validate the image if your microSD card is connected via USB3.

Source: https://developer.nvidia.com/embedded/learn/get-started-jetson-nano-devkit

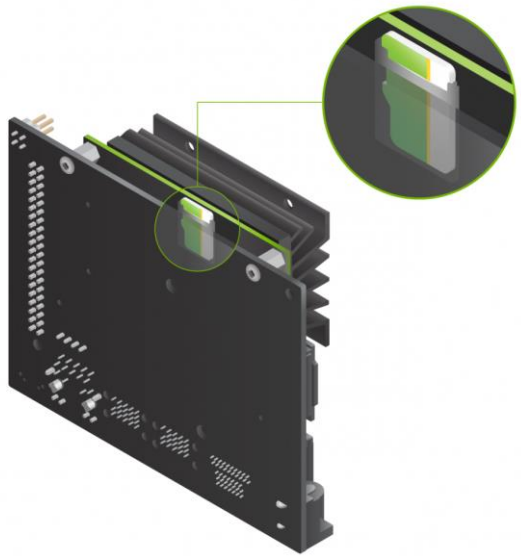
# Software Setup







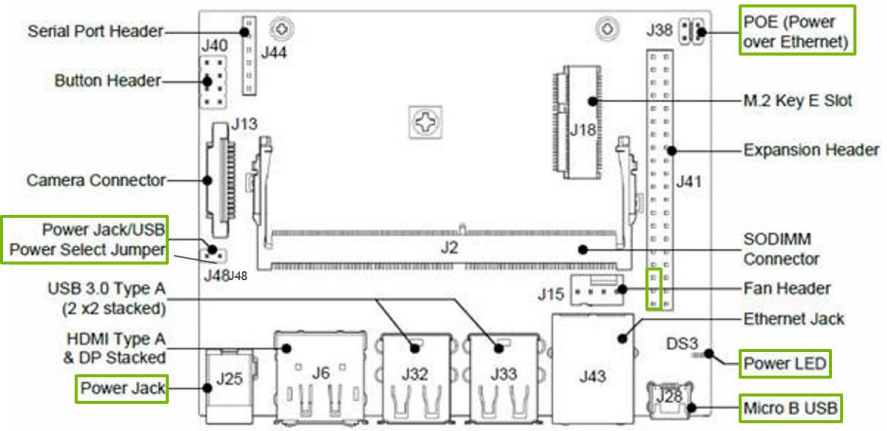
# Software Setup

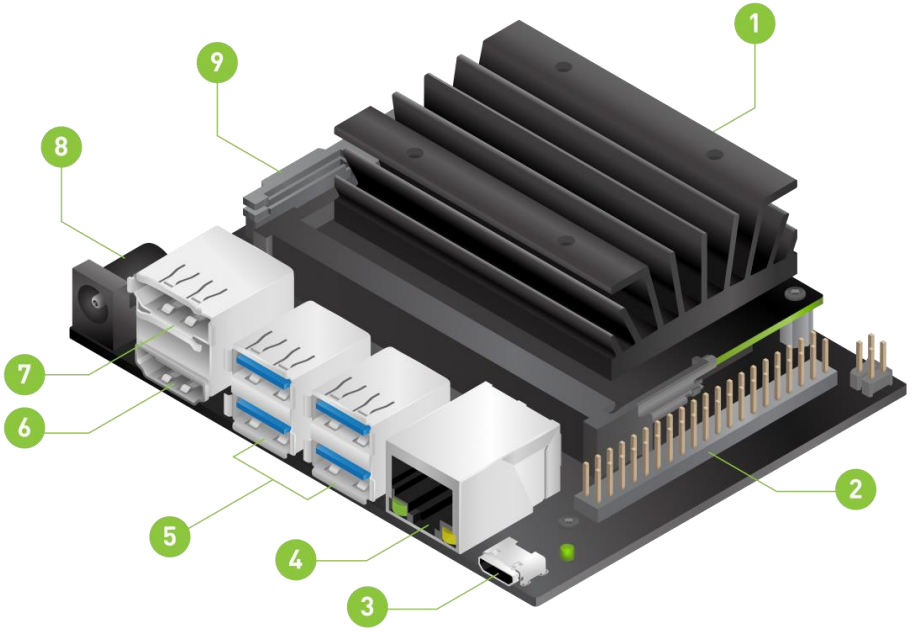


Put Micro-SD card in Jetson Nano
Connect Jetson Nano to peripherals

- Monitor
- Keyboard
- Mouse
- Power – Bridge J48

Follow prompts on screen to setup Linux



# Software Setup

Jetson Nano only has 4GB of memory
A **swap file** is needed
- If sudo permissions are ever denied:
Run:

```
$ sudo -s
```

This will give the user sudo (root) permissions while retaining the current user file location (directory). Enter user's password and continue.

Then swap back to the user with:

```
$ sudo login <username>
```

Enter the user's password and continue

# Software Setup

- Run:
    $ sudo fallocate -l 8G /mnt/8GB.swap
    $ sudo mkswap /mnt/8GB.swap
    $ chmod 0600 /mnt/8GB.swap
    $ sudo swapon /mnt/8GB.swap

j. Once the above is working, add the following line into /etc/fstab:
    $ vim /etc/fstab

- Press “i” key to enter insert mode.
- To the end of the file add:
    “/mnt/8GB.swap  none  swap  sw  0 0”
- Save and Exit (in vi/vim press the “Esc” key to get out of insert mode then type “:wq” and press “Enter”)
- The process for writing to a file through the vim text editor will not change.

k. Reboot the system.
    $ sudo reboot

l. Make sure the swap space gets mounted automatically after reboot:
    $ sudo mkswap /mnt/8GB.swap

- A message about the swap being mounted should appear.

# Software Setup


SSH Setup

- Plug in Wi-Fi Dongle to Jetson Nano
- Connect to preferred wifi network
- Identify Jetson Nano's Local IP Address
  - ifconfig
    - Interface – ethernet (eth0), wireless network (wlan0), loopback (lo)
    - inet <IP Address>
  - Connection Information
    - On the Jetson Nano's desktop screen, find the WiFi connection symbol at the top right corner of the screen (left of the time, sound, etc.).
    - Click on the WiFi symbol and a drop down menu will appear. On this menu click "Connection Information".
    - A window will appear showing you all of the information about the Jetson Nano's "Active Network Connections".
    - Under the section, IPv4, the first item should be the Jetson Nano's IP address.
- SSH from another computer
  - $ ssh <nano username>@<nano IP address>
  - -X argument can be used to open windows on the Nano
    - $ ssh -X …

# Software Setup

Installing TensorFlow
Install prerequisites
$ sudo apt-get install libhdf5-serial-dev hdf5-tools libhdf5-dev zlib1g-dev zip libjpeg8-dev liblapack-dev libblas-dev gfortran

Install and upgrade pip3
$ sudo apt-get install python3-pip
$ sudo pip3 install -U pip testresources setuptools

Install the following python packages
$ sudo pip3 install -U numpy==1.16.1 future==0.17.1 mock==3.0.5 h5py==2.9.0 keras_preprocessing==1.0.5 keras_applications==1.0.8 gast==0.2.2 futures protobuf pybind11

To install TensorFlow 1.15 for JetPack 4.4:
$ sudo pip3 install --pre --extra-index-url https://developer.download.nvidia.com/compute/redist/jp/v44 ‘tensorflow<2’

Source: https://elinux.org/Jetson_Zoo

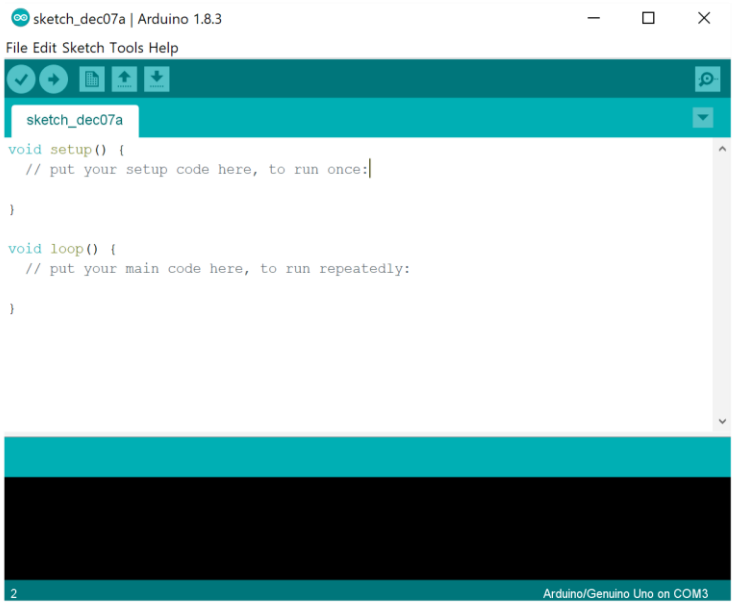
# Software Setup

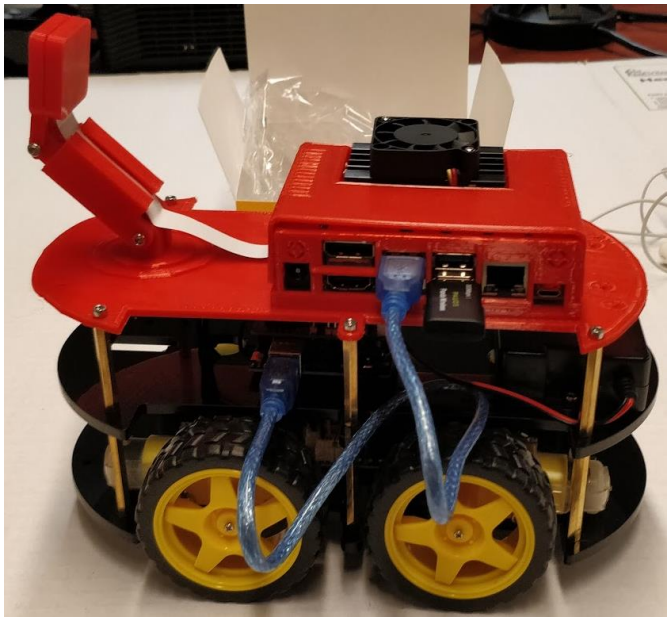
ImageAI install
Run:
    $ pip3 install imageai
PySerial install
Run:
    $ pip3 install pyserial
Test ImageAI and TensorFlow
    $ python3
    > import imageai
    > import tensorflow as tf
    > print(tf.__version__) → 1.15.3
    ctrl-D
If no errors occur, ImageAI and TensorFlow are successfully installed!

ImageAI

# Software Setup

Arduino Setup
Install Arduino IDE on a separate computer from
https://www.arduino.cc/en/Main/Software
Download arduino-code.ino from Lapenna Material/References/Autonomous Vehicle/
Plug the arduino into the computer via the blue USB cable
Open the arduino-code.ino file with the Arduino IDE
Upload the code to the Arduino Uno
Plug the Arduino back into the Autonomous Vehicle

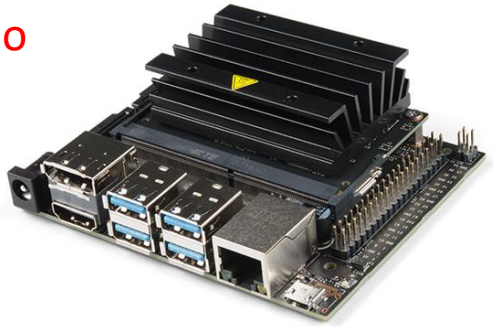


# Software Setup

Jetson Nano and Arduino Setup

*Needs to be done each time you restart the Nano or plug in the arduino



$ ls –l /dev/ttyACM*

Using the correct reference for the arduino

$ sudo chmod a+rw /dev/ttyACM0

Download communicate-with-arduino.py from /Lapenna Material/References/Autonomous Vehicle/ OR git clone https://bitbucket.org/CFDL/opendnnwheel.git

If the arduino is not on port ACM0, you will need to modify the top of this code

```
with serial.Serial('/dev/ttyACM0', 9600, timeout=10) as ser:
    while True:
        …
```

Run the code:

$ python3 communicate-with-arduino.py

# Software Setup

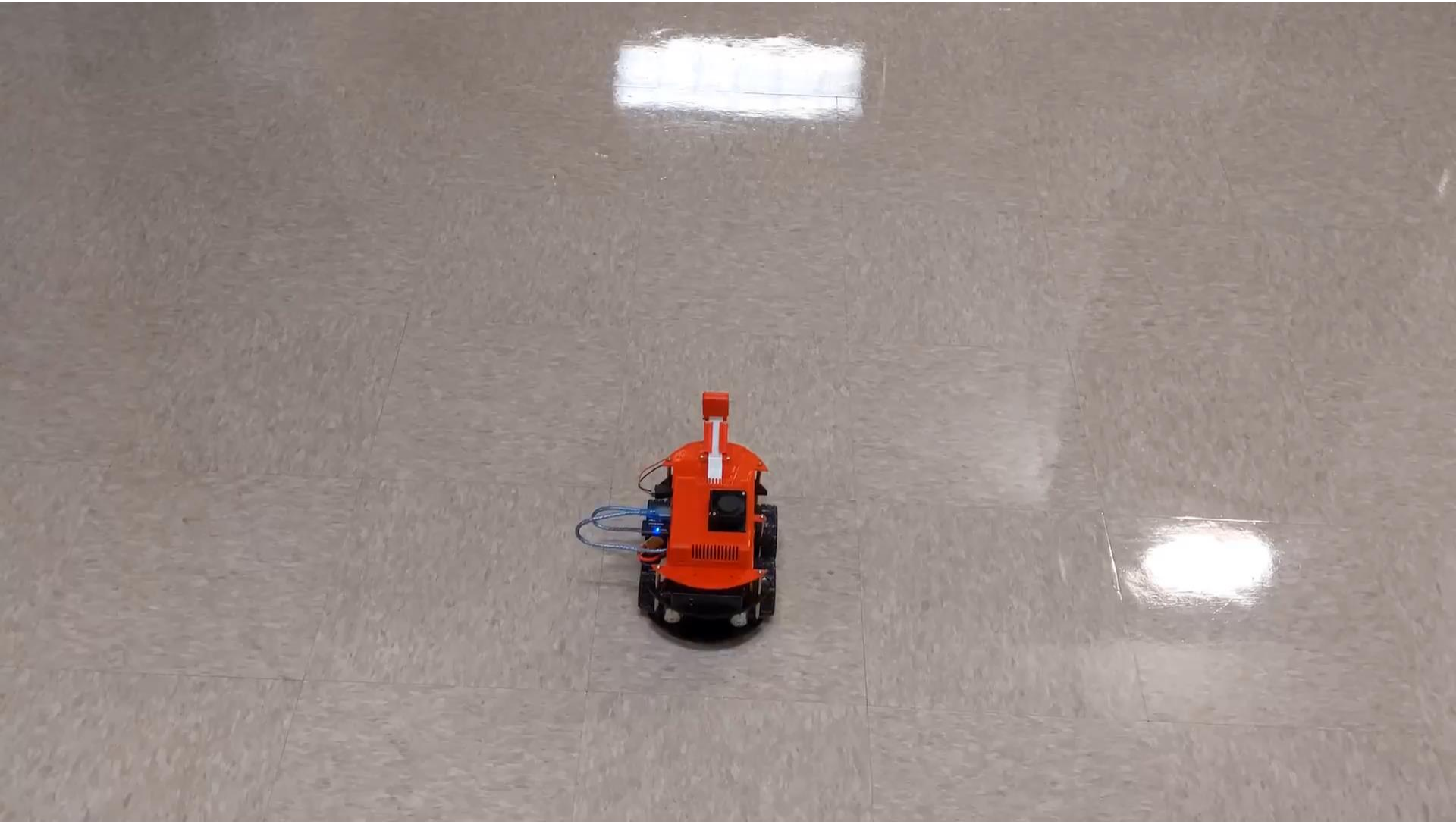
communicate-with-arduino.py
Awaits the user to input character commands

| Command | Action |
|---|---|
| W | Forward |
| S | Backward |
| A | Left |
| D | Right |
| Q | Stop |
| X | Slow Down |
| Z | Speed Up |
| T | Return (Turn 180°) |
| R | Left 90° |
| Y | Right 90° |

*Remember to turn on the battery on the car!

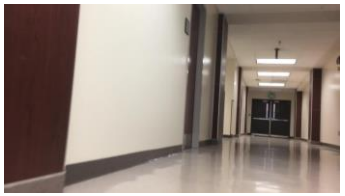

For example, if you wish to go forward, you type ‘w’ or ‘W’ and hit enter.
If wheels turn, you now have a working RC Vehicle!

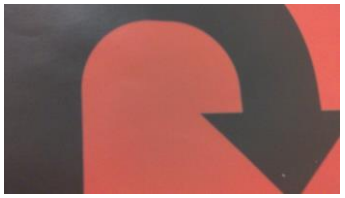
# Software Setup



# Theory

Training
communicate-with-arduino.py
collect-img.py





Terminal — Tue 10:04

Automatic suspend
Computer will suspend very soon because of inactivity.

File Edit View Search Te

```
import numpy as np
import cv2
import os

def gstreamer_pipeline (capture_width=1920, capture_height=1080, display_width=1920, display_heig
ht=1080, framerate=15, flip_method=2) :
    return ('nvarguscamerasrc ! '
    'video/x-raw(memory:NVMM), '
    'width=(int)%d, height=(int)%d, '
    'format=(string)NV12, framerate=(fraction)%d/1 ! '
    'nvvidconv flip-method=%d ! '
    'video/x-raw, width=(int)%d, height=(int)%d, format=(string)BGRx ! '
    'videoconvert ! '
    'video/x-raw, format=(string)BGR ! appsink'  % (capture_width,capture_height,framerate,flip_m
ethod,display_width,display_height))


cap = cv2.VideoCapture(gstreamer_pipeline(flip_method=2), cv2.CAP_GSTREAMER)


num = 0
while os.path.exists('images{}'.format(num)):
    num += 1


# Create file path for images
dirName = 'images{}'.format(num)
os.mkdir(dirName)
print("Directory " , dirName ,  " Created ")

img_num = 0

while(cap.isOpened()):
    ret, frame = cap.read()
    if ret==True:
        cv2.imwrite('%s/image%d.jpg' % (dirName,img_num),frame)
        print('frame captured%d' % (img_num))
        img_num += 1
        # Specifies display size
        #display = cv2.resize(frame,(640,480))
        #cv2.imshow('display',display)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            print('Pressed Q')
            break
    else:
        break
```

-- INSERT -- 28,1 Top

File Edit View Search Termi

```
captured image465.jpg
captured image466.jpg
captured image467.jpg
captured image468.jpg
captured image469.jpg
captured image470.jpg
captured image471.jpg
captured image472.jpg
captured image473.jpg
captured image474.jpg
captured image475.jpg
captured image476.jpg
captured image477.jpg
captured image478.jpg
captured image479.jpg
captured image480.jpg
```

# Theory

Training

scp <nano username>@<nano IP address>:/path/to/images/*.jpg <destination>

Images

Sort Images to classes
Compress to .tar

Turn on public link sharing
Identify code in link
“drive.google.com/file/d/<code>/view”

Images

Download using gdown <code>

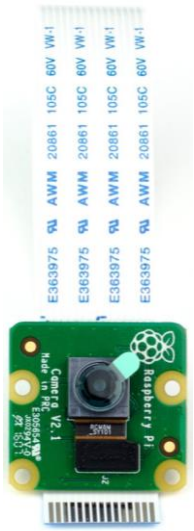
Google Drive

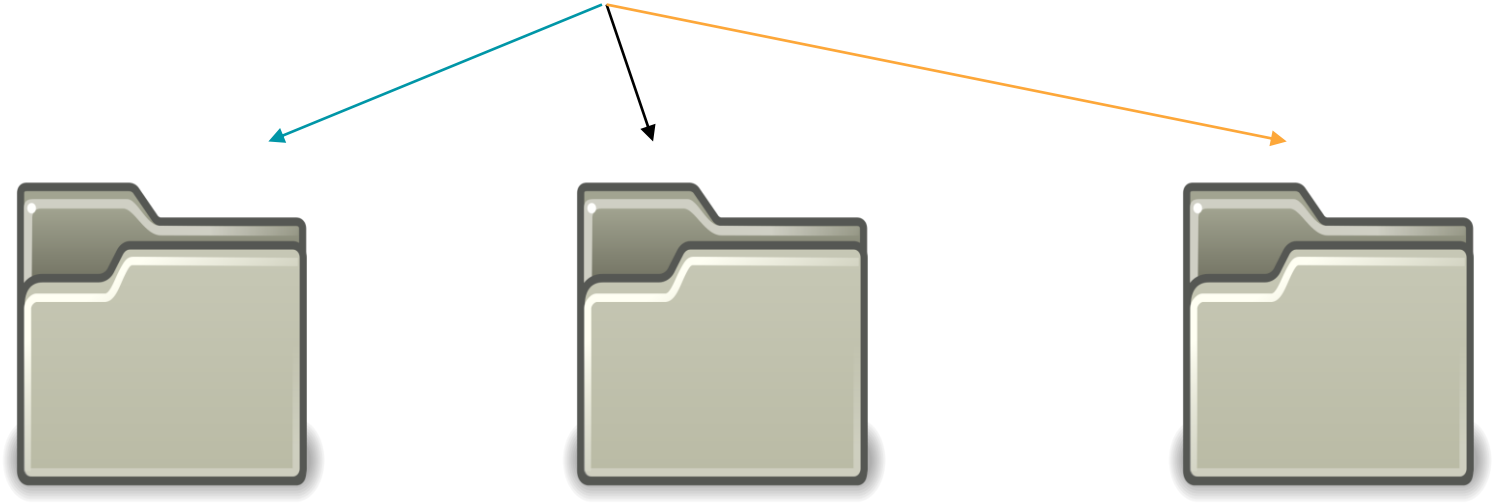

Complete Training

# Autonomous Vehicle

## Input / Output

- ✓ Raspberry Pi Camera - Input: 1920(px)x1080(px)x3(RGB) = 6,220,800
- ✓ Capture many images using the car. (Getting as close as possible to the environment that you will run the neural network in)
- ✓ image0.jpg, image1.jpg, … , image(n).jpg
- ✓ Create directories for each class



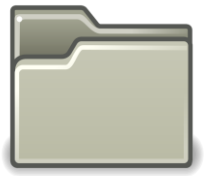


W
(forward)

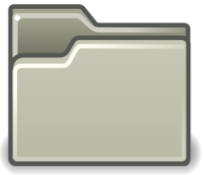
L
(correct left)

R
(correct right)

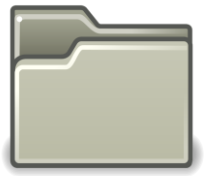
# Autonomous Vehicle



W
(forward)



L
(correct left)

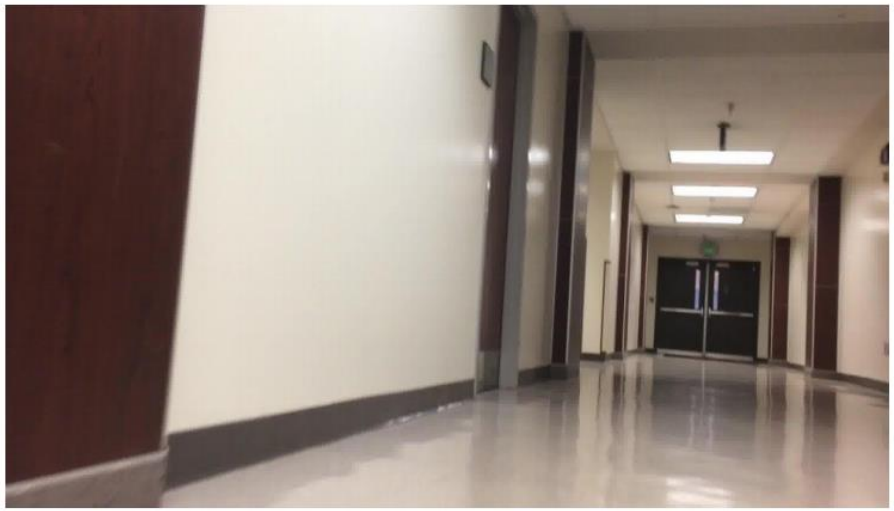


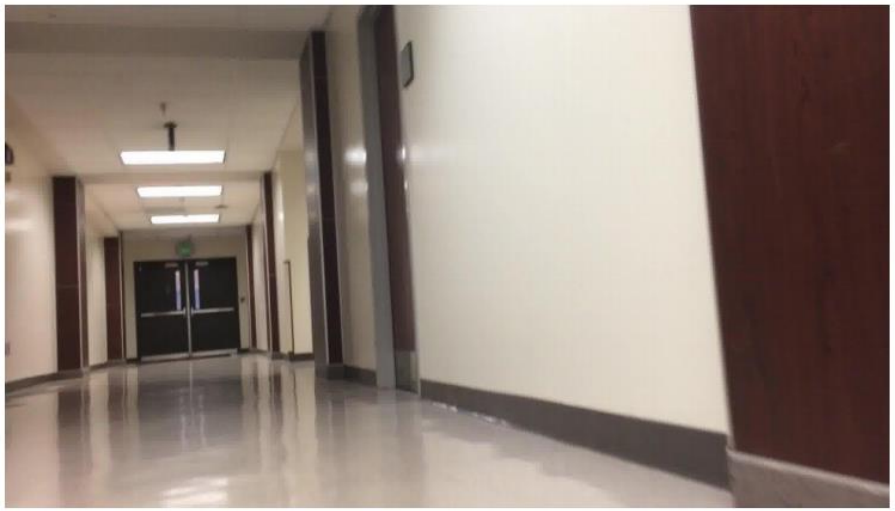
R
(correct right)

Sort images into their respective folders (time consuming)

- W – images in which the car needs to continue going straight
- L – images in which the car needs to turn left slightly to avoid the right wall
- R – images in which the car needs to turn right slightly to avoid the left wall

Data Augmentation : flip, rotate images





# Autonomous Vehicle

## Training the Model

Downloading from Google Drive:

```
$ conda create –n gdown
$ conda activate gdown
$ conda install –c conda-forge gdown
$ gdown <drive link>
```

ImageAI

train.tar: https://drive.google.com/uc?id=1xQ0vzz43s2M7ViFksds5dWezErXVI5m

test.tar: https://drive.google.com/uc?id=1ivTvYwZT5wlDp7fAwxAwM4NOt04LgbJx

model_train.py: https://drive.google.com/uc?id=1UU7hZvcoFGoJL-gSNh-VqQEkxHmlFUel

https://imageai.readthedocs.io/en/latest/

Extract all files and put the train and test folder within a folder called image_labels

Install ImageAI, Keras, TensorFlow-gpu, and opencv

- conda install -c powerai imageai
- conda install tensorflow-gpu==1.13.1

# Autonomous Vehicle

## Training the Model

**Training: python model_train.py**

```
model_train.py
from imageai.Prediction.Custom import ModelTraining
from imageai.Prediction import ImagePrediction
import os

model_trainer = ModelTraining()
model_trainer.setModelTypeAsResNet()

model_trainer.setDataDirectory("image_labels")

model_trainer.trainModel(num_objects=3, num_experiments=5, enhance_data=False,
batch_size=20, show_network_summary=True)
```

**Training output: model.h5(~90MB), model_class.json**

```
image_labels/models/
image_labels/json/
```

ImageAI

https://imageai.readthedocs.io/en/latest/

# Autonomous Vehicle

## Training the Model

| Layer (type) | Output Shape | Param # | Connected to |
|---|---|---|---|
| input_1 (InputLayer) | (None, 224, 224, 3) | 0 | |
| conv2d (Conv2D) | (None, 112, 112, 64) | 9472 | input_1[0][0] |
| batch_normalization_v1 (BatchNo | (None, 112, 112, 64) | 256 | conv2d[0][0] |
| activation (Activation) | (None, 112, 112, 64) | 0 | batch_normalization_v1[0][0] |
| max_pooling2d (MaxPooling2D) | (None, 55, 55, 64) | 0 | activation[0][0] |
| conv2d_2 (Conv2D) | (None, 55, 55, 64) | 4160 | max_pooling2d[0][0] |
| batch_normalization_v1_2 (Batch | (None, 55, 55, 64) | 256 | conv2d_2[0][0] |
| activation_1 (Activation) | (None, 55, 55, 64) | 0 | batch_normalization_v1_2[0][0] |
| conv2d_3 (Conv2D) | (None, 55, 55, 64) | 36928 | activation_1[0][0] |
| batch_normalization_v1_3 (Batch | (None, 55, 55, 64) | 256 | conv2d_3[0][0] |
| activation_2 (Activation) | (None, 55, 55, 64) | 0 | batch_normalization_v1_3[0][0] |

. . .

ImageAI

https://imageai.readthedocs.io/en/latest/

# Autonomous Vehicle

## Training the Model

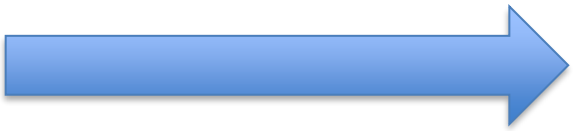
```
Total params: 23,593,859
Trainable params: 23,540,739
Non-trainable params: 53,120
_________________________________________________________________
Found 7213 images belonging to 3 classes.
Found 1340 images belonging to 3 classes.
JSON Mapping for the model classes saved to  image_labels/json/model_class.json
Number of experiments (Epochs) :  5
. . .
Epoch 1/5
2020-09-25 15:27:14.849756: I tensorflow/stream_executor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
67/67 [==============================] - 70s 1s/step - loss: 4.9358 - acc: 0.5813
Epoch 00001: val_acc improved from -inf to 0.58134, saving model to image_labels/models/model_ex-001_acc-0.581343.h5
361/361 [==============================] - 445s 1s/step - loss: 0.6167 - acc: 0.8099 - val_loss: 4.9358 - val_acc: 0.5813
. . .
Epoch 5/5
67/67 [==============================] - 31s 458ms/step - loss: 0.2381 - acc: 0.9231
Epoch 00005: val_acc improved from 0.91791 to 0.92313, saving model to image_labels/models/model_ex-005_acc-0.923134.h5
361/361 [==============================] - 190s 526ms/step - loss: 0.0680 - acc: 0.9750 - val_loss: 0.2381 - val_acc: 0.9231

model_ex-005_acc-0.923134.h5 - 94MB - 92.31%
```

# Running the Autonomous Vehicle

Autonomous Vehicle Inference

model.h5 model_class.json



autonomous-car.py

add location of .h5 and .json files

$ python3 autonomous-car.py

model.h5

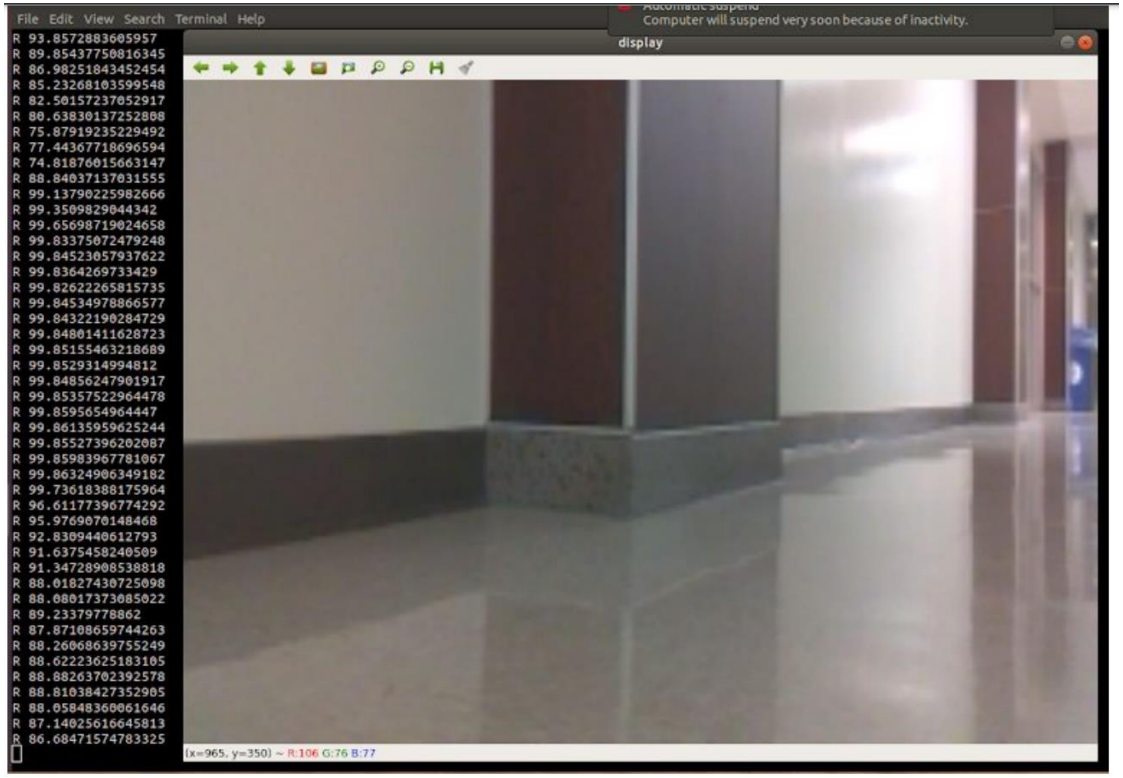

# Autonomous Vehicle

## Inference on the Jetson Nano

The .h5 and .json file are stored on the Jetson Nano
The file location is input in the python code:

```
execution_path = os.getcwd()
prediction.setModelPath(os.path.join(execution_path, "7-30-test/model.h5") )
prediction.setJsonPath( os.path.join(execution_path, "7-30-test/model_class.json") )
```

Then the Jetson Nano writes an image to image.jpg, and makes a prediction based on that image
This happens multiple times per second in a loop to allow the car to drive



Joint Institute for Computational Sciences

# Thank you!

INNOVATIVE COMPUTING LABORATORY