Operation Manual
C.I.A. Team 503: 1/5th Hardware in Loop

Richard Allen
Kathleen Bodden
David Gordon
Micah Hilliard
Chet Iwuagwu
Nicholas Muoio

FAMU-FSU College of Engineering
Florida State University

EML 4552C Senior Design 2
Dr. Shayne McConomy
March 03, 2023

Table of Contents

List of Figures

Project Overview

The shift to autonomy is a primary focus for producers moving forward. Researchers are finding ways to make driverless cars while battling energy consumption. Our team aims to design a small-scale autonomous car that reduces energy losses by keeping a constant speed.

Scaling down the project is beneficial to our goals. We can easily deal with challenges that are harder to test with a larger model. We can also replicate a real environment with software that interacts with the scaled car. This is called hardware-in-the-loop. Hardware-in-the-loop is a testing technique that reproduces different environments based on physical signals.

We split the project into two sides. These include the mechanical side of the project and the vision side. The two large sides of the project were broken even further into four parts. Looking at smaller parts of our project allows us to analyze the more complex design. The four parts include object detection, path planning, the steering, and the motor. Our team focuses on the mechanical side of the project while a separate team focuses on the vision side. Communication with the vision team is important to the success of the project. The mechanical side includes a steering design that rotates to a set angle. The angle is provided by the vision team, to adjust the direction the car is heading. The mechanical side also includes a motor design working to propel the car forward.

The design should work without actions from a user. The path for the car goes from the back entrance of the Aero-Propulsion, Mechatronics and Energy Building to the B side entrance of the College of Engineering. The project will be successful when efficient power consumption is achieved. This will happen when less energy is lost from the car's movement.

Component Module Description

1. Radio Flyer Tesla Model S Power wheel

Model No: 910
VIN: 5RFS11DD4GKP43639
Power Supply: Lithium Ion 14.4V, 13.2Ah, 190Wh
Charge Time: 4hr 30min

FIGURE 1: TESLA MODEL S (UNMODIFIED)

FIGURE 2: LITHIUM-ION PREMIUM BATTERY

General Safety Guidelines

1. Before children use this vehicle, an adult should carefully evaluate the driving area as well as the child's ability to drive the vehicle safely. Teach appropriate safety rules to your child before allowing operation of this vehicle. These rules should also be reviewed with other children who want to drive the vehicle.
2. Remain seated. A child who is not sitting in the seat when the vehicle is in motion could fall or cause the vehicle to tip over.
3. DO NOT operate the vehicle at night.
4. DO NOT operate the vehicle near steep inclines as it can cause the following:
    a. Unsafe speed, even if the pedal is released.
    b. Tipping
    c. Loss of traction, causing the vehicle to slip.
    d. Rolling backwards at an unsafe speed.
5. Avoid contact with moving/rotating parts, such as the motor, gear box, and wheels. Contact with these parts can cause serious injury.
6. DO NOT operate the vehicle when it's positioned on its side or upside-down.
7. DO NOT operate the vehicle near flammable vapors (gasoline, paint thinner, acetone, liquid wax, etc.) as the electrical switch emits an internal spark, which could cause an explosion or fire.
8. To prevent unsupervised use of the vehicle, remove the battery when not in use.
9. It is the responsibility of the adult who assembled this product to properly install all parts included in the carton. These instructions are valuable.
10. Check nuts and bolts often and tighten if necessary.
11. Any parts showing evidence of wear should be replaced immediately.
12. Check all screws and their protective coverings regularly and tighten as required. Check plastic parts on a regular basis for cracks or broken pieces.
13. During snowy or rainy weather, the vehicle should be stored inside.
14. DO NOT operate the vehicle in wet or snowy conditions.
15. DO NOT clean the vehicle by spraying it with a hose. Water or moisture in the motors, battery, or electrical components can cause component failure.
16. To clean the vehicle, use a dry cloth. A non-wax furniture polish can be used to clean plastic parts. Do not spray the vehicle with a hose or submerge in water.
17. DO NOT use the charger for any other products. It is intended for Radio Flyer 14.4V lithium-ion battery powered vehicles only.

18. DO NOT operate the charger if it has been impacted, dropped, or damaged in any way. Examine the charger and battery before each use and replace if wear or damage is found.
19. The charger is designed to operate on standard household electrical power (110/120 VDC). Do not attempt to use the charger on any other voltage level!
20. Never modify the electrical system as this may result in a risk of electric shock, electrocution, or fire.
21. To reduce risk of damage to the charger, pull from the plug rather than the cord when removing from an outlet.
22. Make sure the cord is located so that it will not be stepped on, tripped over, or otherwise subjected to damage or stress.
23. An extension cord should not be used unless necessary. Use of an improper extension cord could result in the risk of fire, electric shock, or electrocution.
24. To prevent electric shock, do not immerse the charger or battery in water when cleaning. To clean, remove the charger/battery plug from the outlet and wipe with a dry cloth.
25. DO NOT place any object on top of the charger or place the charger on a soft surface that may result in excessive heat. Place the charger away from any heat sources.
26. When charging is complete, disconnect the charger from the wall outlet and the battery. Store the battery and charger in a dry place out of reach from children.
27. DO NOT disassemble the charger or attempt to repair a damaged charger. If the charger is damaged contact Radio Flyer customer service.

Controls

1. POWER BUTTON: Activates unit when pedal is pressed. If the pedal is pressed during operation, the unit stops.
2. See Appendix B for detailed Tesla Instruction Manual.

2. Steering Actuation

General Safety Guidelines

Read all the instructions before using the Steering Actuation:

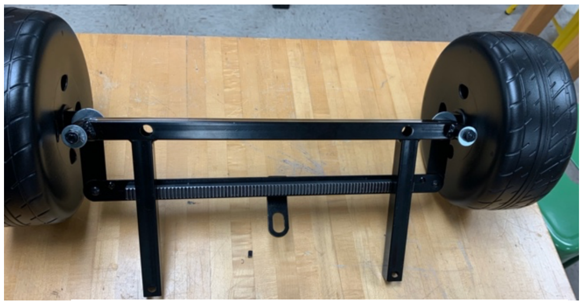


FIGURE 3: STEERING LINKAGE (DETACHED)

1. Use the steering actuation only as described in this manual. Anything outside the guidelines are not recommended, may cause injury to persons.
2. Rotate steering wheel to determine if rotary motion is difficult. If rotation is choppy, apply gear grease directly to pinion.
3. Be careful not to get gear grease on the electrical components of the motor.
4. DO NOT place your hand on the rack when the steering is in motion. This may lead to personal injury.
5. DO NOT rub your hand against the rack. This may lead to personal injury.

6. DO NOT hold the pinion if the motor is in motion. This may lead to personal injury.
7. DO NOT place your hand near the tires while the vehicle is in motion. This may lead to personal injury.

Controls

1. **Code Prototype:** The code provided in Appendix D demonstrates controlling a DC motor with an encoder. The code provided is useful in finding the position of the DC Motor Shaft, which in turn can output the position of the pinion on the steering rack. In the real-life application, the motor will have to undergo a calibration phase as encoders are incremental and not based on specific positions. After calibration, the DC Motor will position itself at the center of the rack, to ensure the wheels are pointing forward. After the position is secured, data will be read in from the NVIDIA Jetson, in the form of a desired shaft angle/steering angle. The code will use a Proportional-Derivative (PD) controller to ensure correct motor positioning.
2. START: While receiving data from the NVIDIA Jetson, the steering will move automatically to follow a prescribed path once the pedal is pressed. Without data input from the sensors, the motor will move the rack to the extreme position to cause circular motion.
3. EMERGENCY HANDLING: If the user chooses to manually control the direction, grab hold of the steering wheel and turn where needed.
4. STOP: When connected to the NVIDIA Jetson, the vehicle will stop once it reaches its destination. Otherwise, the user needs only remove their foot from the pedal to cut the power to the system.

3. Motor Actuation

WARNING: TO AVOID THE RISK OF ELECTRICAL SHOCK, FIRE, OR INJURY, ALWAYS MAKE SURE THE PRODUCT IS UNPLUGGED FROM THE ELECTRICAL OUTLET BEFORE RELOCATING, SERVICING OR CLEANING IT.

General Safety Guidelines

1. To access the motors, remove the seat from the vehicle.
2. DO NOT interfere with motor wiring. This may lead to electrocution and personal injury.

Controls

1. START: Upon integration with team 504, once the pedal is pressed, the rear motors will turn at a PWM signal set by team 504 serial outputs. Without integration, the rear motors will move at a constant PWM signal for a constant velocity.
2. EMERGENCY HANDLING: If the user chooses to stop the vehicle, remove foot from pedal.

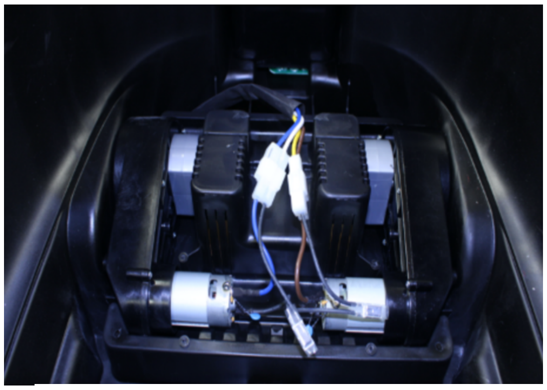


FIGURE 4: REAR MOTOR SETUP

3. STOP: If integrated with team 504, the vehicle will stop once it reaches its destination. Otherwise, the user needs only remove their foot from the pedal to cut the power to the system.

Integration of Components

Material List

Link to final bill of materials: Bill Of Materials.xlsx

Wiring Diagrams

See Appendix A.

Assembly Process

1. Align extruding edges of rims to the slots in the tire until you hear a snap sound. This means the rim has been placed successfully.
2. Connect the steering connector from wheel to steering column until you hear a clicking sound.
3. Align the steering wheel with the column such that the hole in the column stays in the support at the bottom of the wheel.
4. Align holes from steering wheel to column and install M5x65 bolt.
5. Insert the seat into the chassis and lock it back until you hear a clicking sound. The extrusions of the seat should fit into the cut outs in the chassis near the motors.
6. For battery installation, open the trunk and insert the battery vertically in the battery slot until you hear a clicking sound.
7. Using four ¼-20x3/8 screws, attach the rack to the linkage by aligning the respective holes.
8. Place the motor in the motor housing on linkage.
9. Align the rack such that the steering is straight.
10. Attach the shaft with the pinion to the motor shaft. The pinion grooves should align with the grooves of the steering rack.
11. Connect teensy microcontroller in the frunk to the connector coming from behind the front wheels.
12. See Appendix B for assembly guide.

Other Components

- NVIDIA Jetson (Figure 13)
- Intel RealSense Depth camera
    - The initial camera design surrounds the camera and places it inside the front grill of the car. Holes are needed to be cut into the grill so that the 3D printed part shown in figure 15, located in Appendix C, can be screwed into the grill and the camera can be secured in a discrete manner. This design was no longer sufficient due to the new camera being larger.
- Emergency switch box
- Power distribution block
- Wiring of each of the components

Operation (Integrated)

1. Ensure a charged battery is properly connected in the trunk of the vehicle.
2. Place a child in the seat of the vehicle. The child should remain seated during the duration of the course.
3. Press pedal to begin the course. Vehicle motion will start and follow a path away from the college of engineering.
4. Using path data from the NVIDIA Jetson, the vehicle will reach its destination and stop at the AME building. The prescribed velocity comes from PWM signals, sent out from the Teensy microcontroller based on the outputs from the NVIDIA Jetson (See the operation manual for team 504 for sensor and GPU operation).
5. Remove the battery from the trunk to power off the vehicle.
6. Remove the child from the seat of the vehicle.

Operation (Not Integrated)

1. Ensure a charged battery is properly connected in the trunk of the vehicle.
2. Press pedal to begin the course. Vehicle motion will start and follow a circular path.
3. After a set time, the vehicle motion will stop.
4. Remove the battery from the trunk to power off the vehicle.
5. Remove the child from the seat of the vehicle.

Related Documents

The important documents and files related to this project's completion can be found on the Team 503 senior design webpage. The link can be found below:

https://web1.eng.famu.fsu.edu/me/senior_design/2023/team503/

Team 503: Contact Information

The following is the group members' contact information:

Richard Allen
Phone Number: (954)415-2063
Email: richard3.allen@famu.edu
Linkedin: https://www.linkedin.com/in/richard-alleniv/

Kathleen Bodden
Phone Number: (786)613-0920
Email: kmb19b@fsu.edu
Linkedin: https://www.linkedin.com/in/kathleen-bodden-8a9045207/

David Gordon
Email: David2.gordon@famu.edu
Linkedin: https://www.linkedin.com/in/david-gordon-61aa62252/

Micah Hilliard
Phone Number: (386)249-3474
Email: mjh18e@fsu.edu
Linkedin: https://www.linkedin.com/in/micah-hilliard/

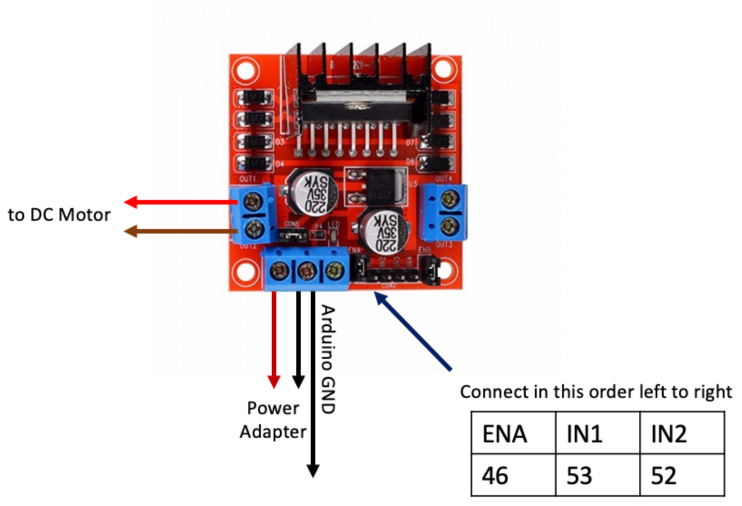
Chet Iwuagwu
Phone Number: (850)321-9576
Email: ciwuagwu@fsu.edu
Linkedin: https://www.linkedin.com/in/chetanna-iwuagwu-313886143/

Nicholas Muoio
Phone Number: (561)289-4317
Email: nlm19b@fsu.edu
Linkedin: https://www.linkedin.com/in/nich-muoio-908b07252/

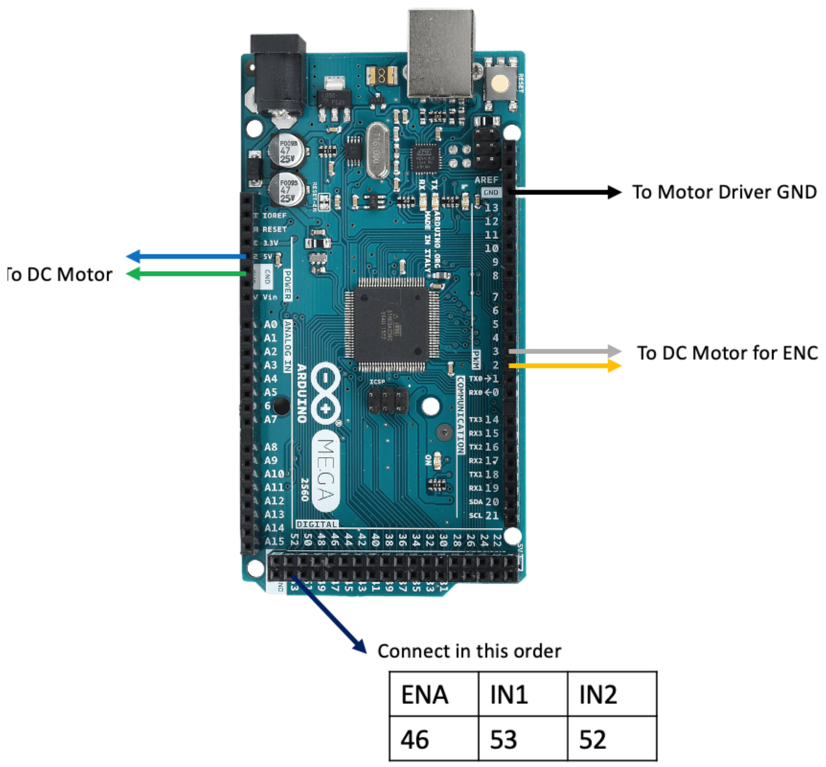
Troubleshooting

Issue	Possible Cause(s)	Corrective Action(s)
1. Tesla Power wheel		
Car doesn't run	Dead battery	Replace battery and put current battery to charge. You can also charge the current battery while it remains in the vehicle. Vehicle cannot operate while it charges.
Horn and headlights don't work	Dead battery	Try charging the battery inside or outside the vehicle.
Car doesn't run	Buttons are not fully pressed	Make sure the forward button is fully depressed, as well as the pedal.
Battery does not light up.	Wires are not fully connected.	Check connections A, B, C, and D. Connection A located at battery. Connection B located under rear panel. Connection C located behind front wheels. Connection D located at rear motors.
Vehicle won't assemble correctly	Improper tool	Phillips head tool driver
Vehicle won't reverse	Wrong button pressed	Switch the button from forward to reverse.
2. Steering Actuation		
Staggered rotation	Wear on rack or pinion.	Apply gear grease directly on pinion.
Pseudocode won't run	Libraries are not installed, or mechanical setup is incorrect.	Using Arduino IDE, download the appropriate libraries. Follow wiring diagram for mechanical setup.
Wheels don't change direction	Steering wheel shaft and/or rack and pinion may not be connected to steering linkage.	Insert steering shaft into linkage. Change linkage direction and align pinion on rack.
3. Motor Actuation		
Motors won't rotate	Wires are not fully connected.	Check connections from motor into microcontroller and motor driver(s).

Appendix A: Connections and Wiring Diagrams



ENA	IN1	IN2
46	53	52



ENA	IN1	IN2
46	53	52

FIGURE 5: DC MOTOR CONNECTIONS

Appendix B: Tesla Instruction Manual

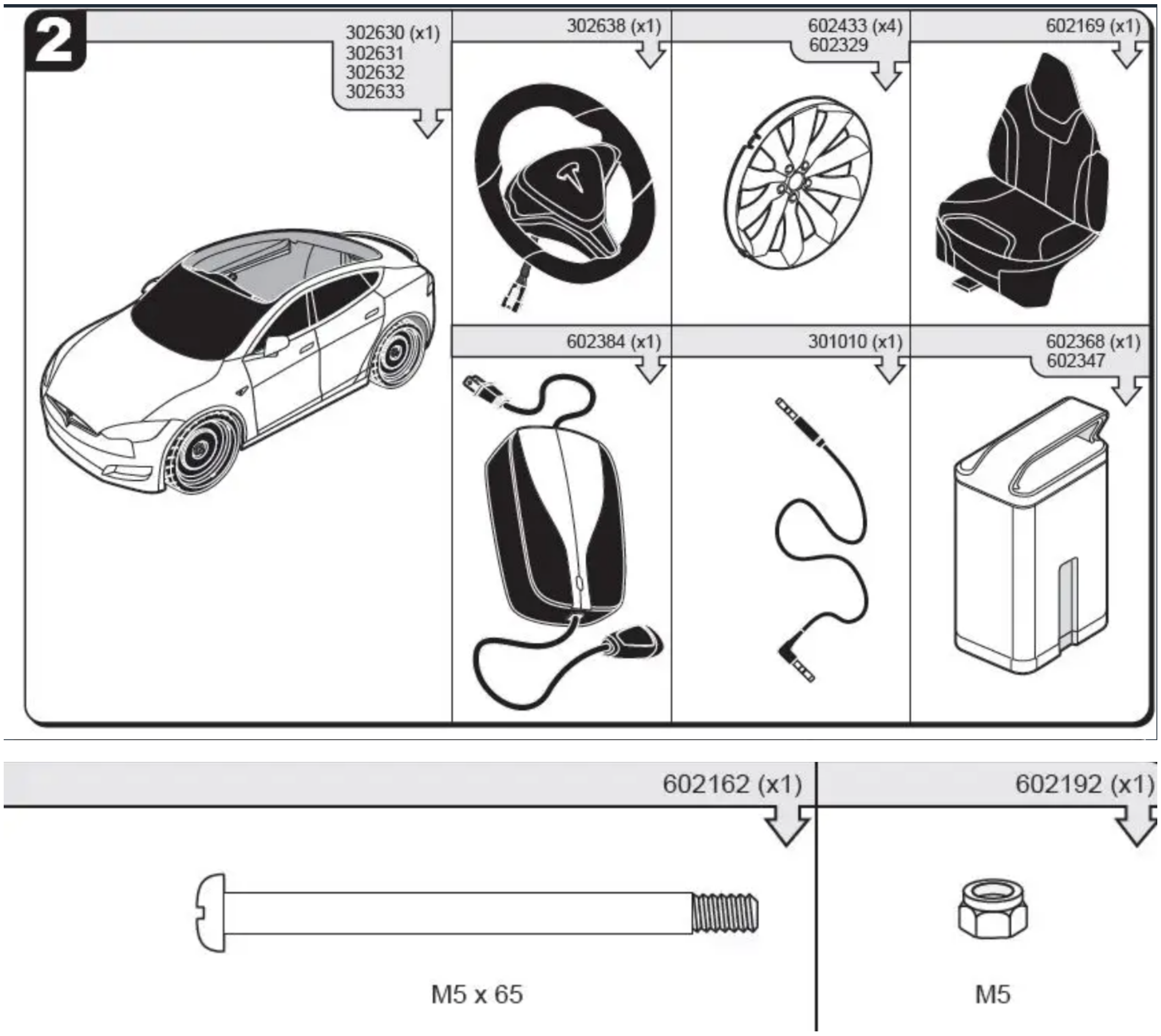
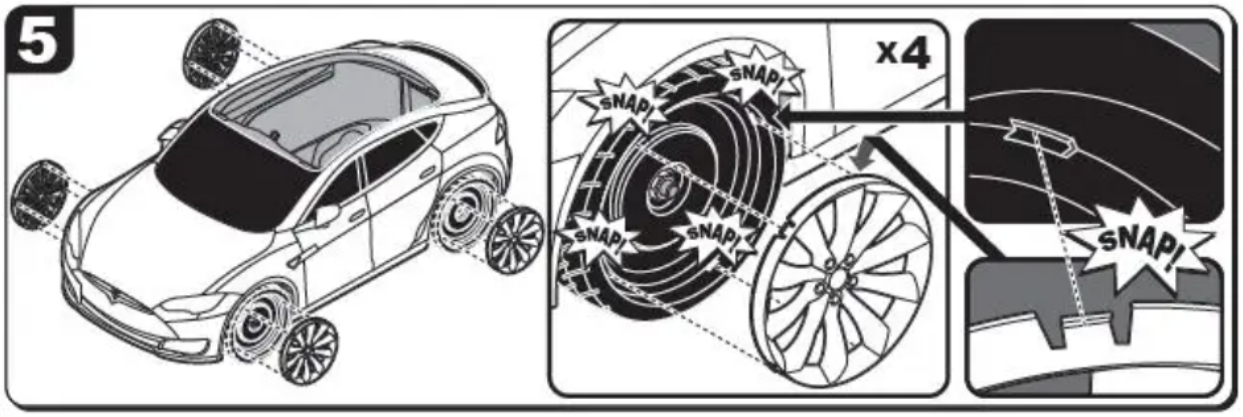
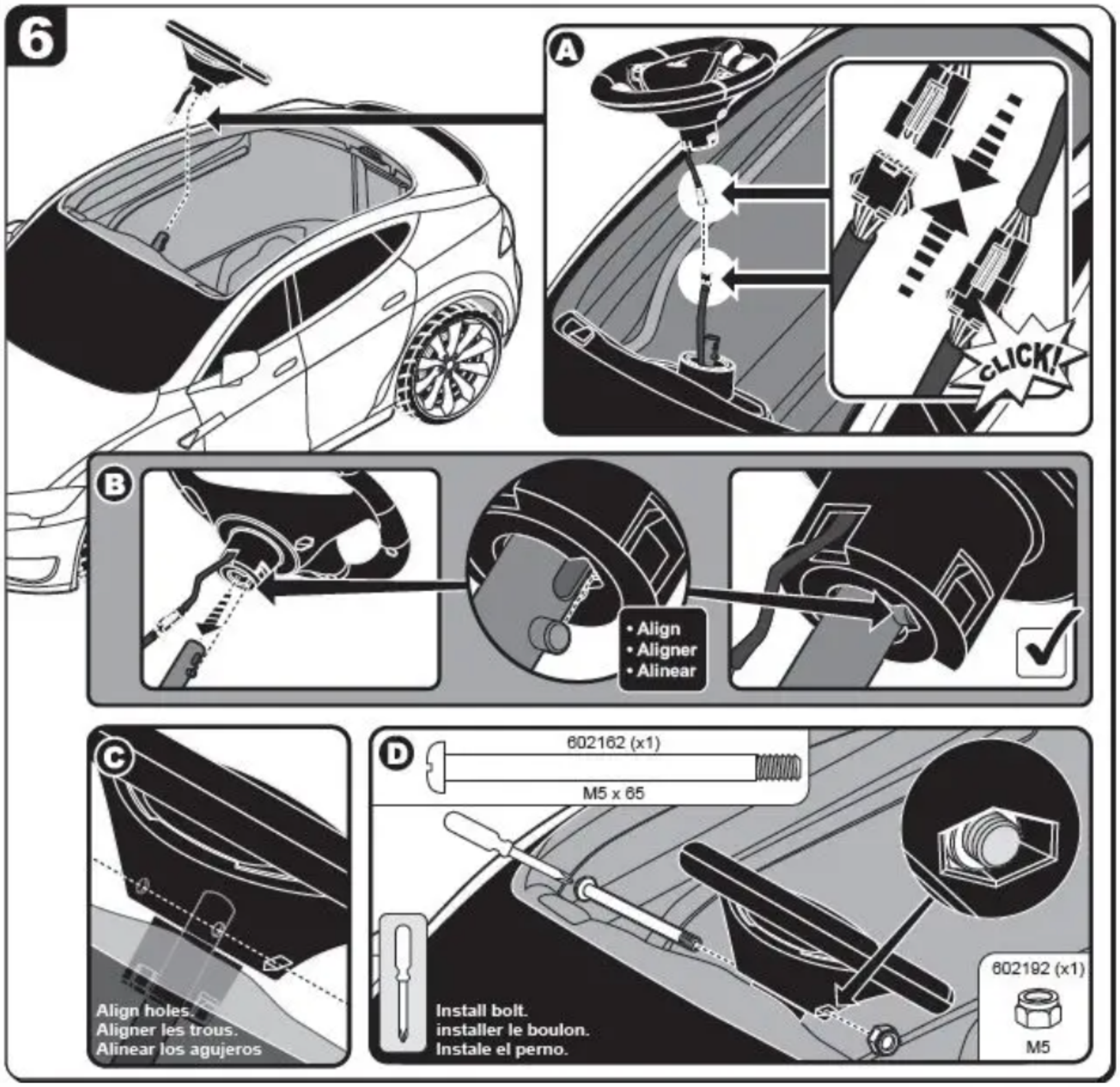


FIGURE 6: PARTS


5
x4
SNAP!
SNAP!
SNAP!
SNAP!
SNAP!


6
A
CLICK!
B
• Align
• Aligner
• Alinear
C
Align holes.
Aligner les trous.
Alinear los agujeros
D
602162 (x1)
M5 x 65
Install bolt.
installer le boulon.
Instale el perno.
602192 (x1)
M5

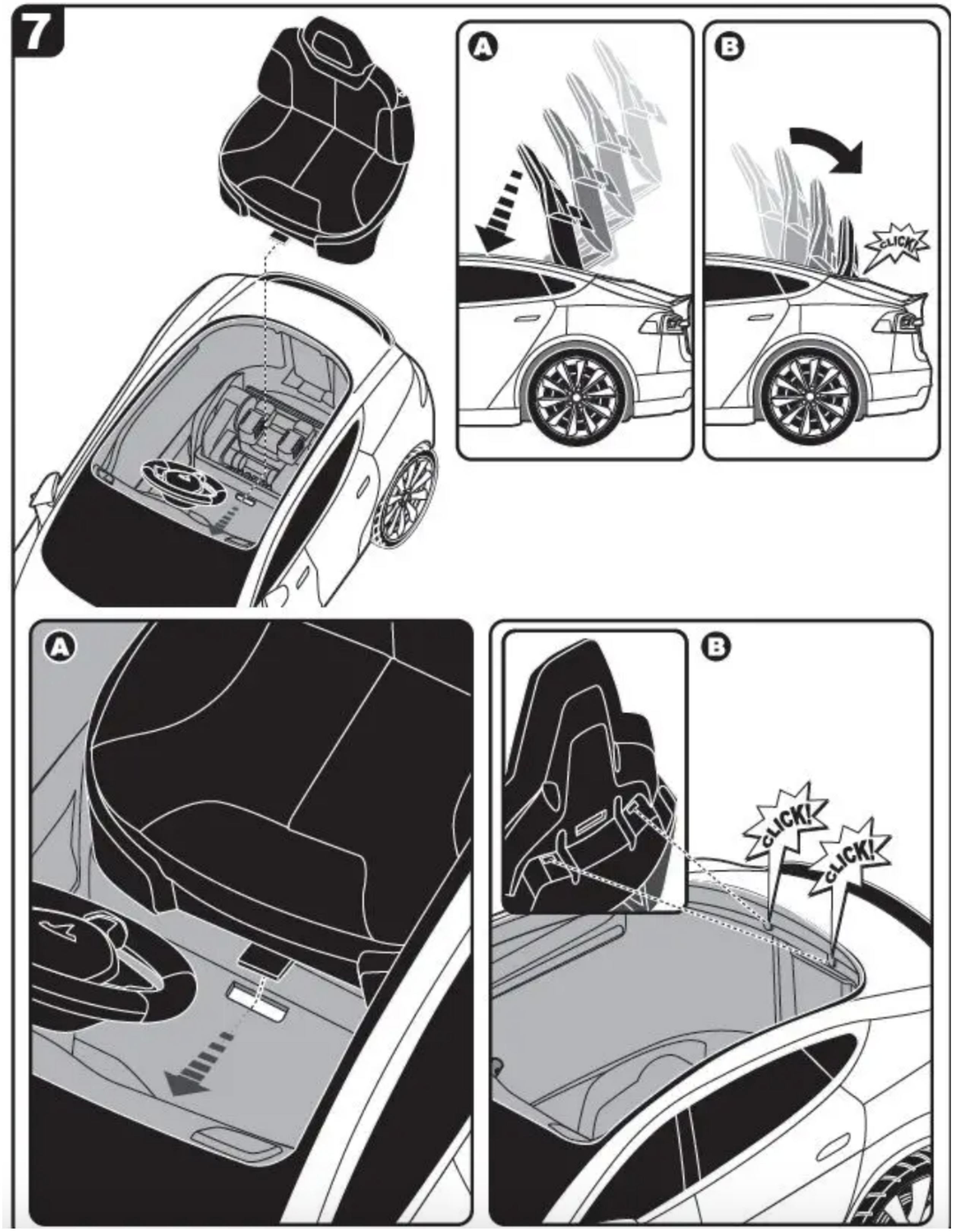


FIGURE 7: ASSEMBLY

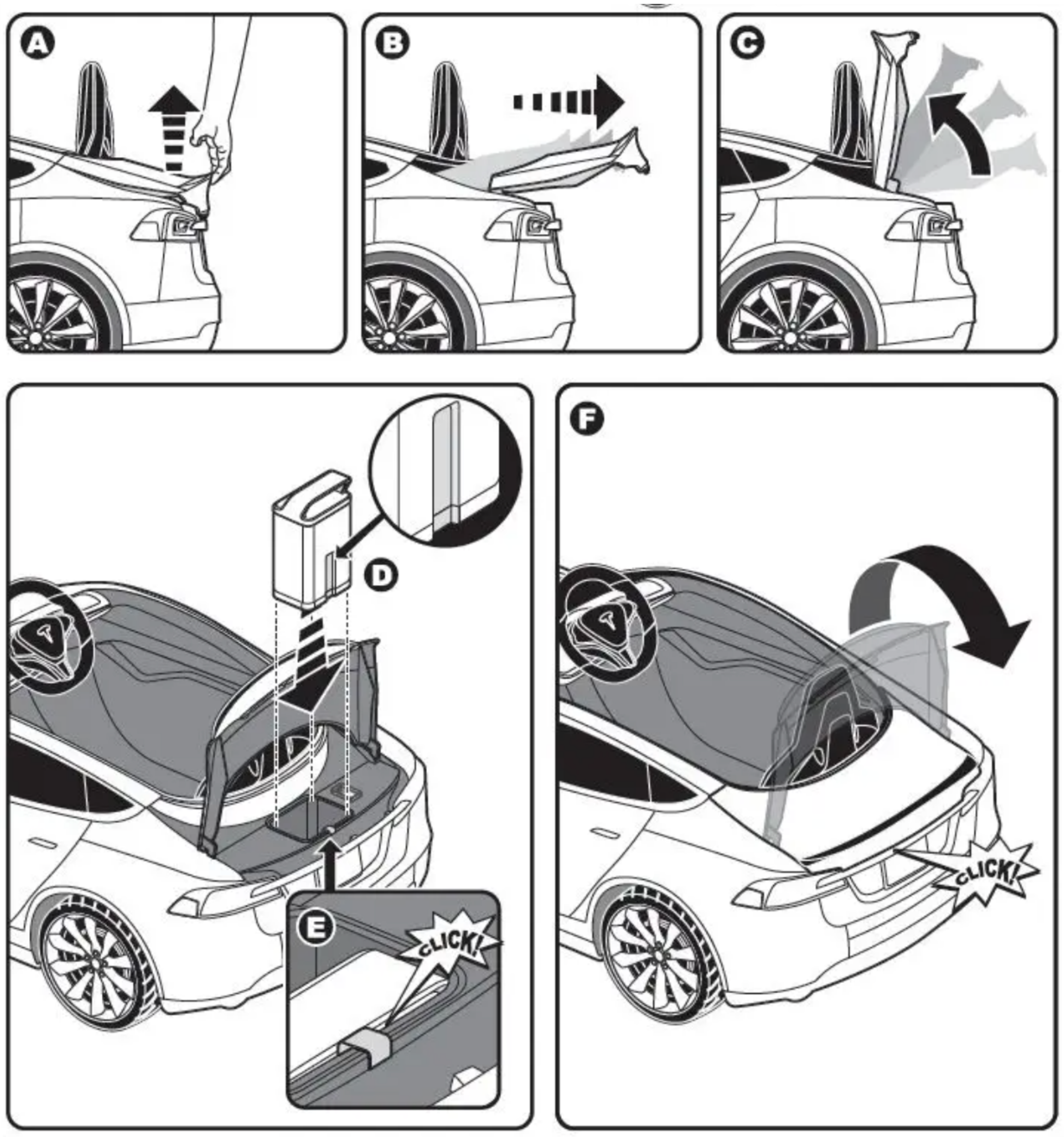


FIGURE 8: BATTERY INSTALLATION & REMOVAL

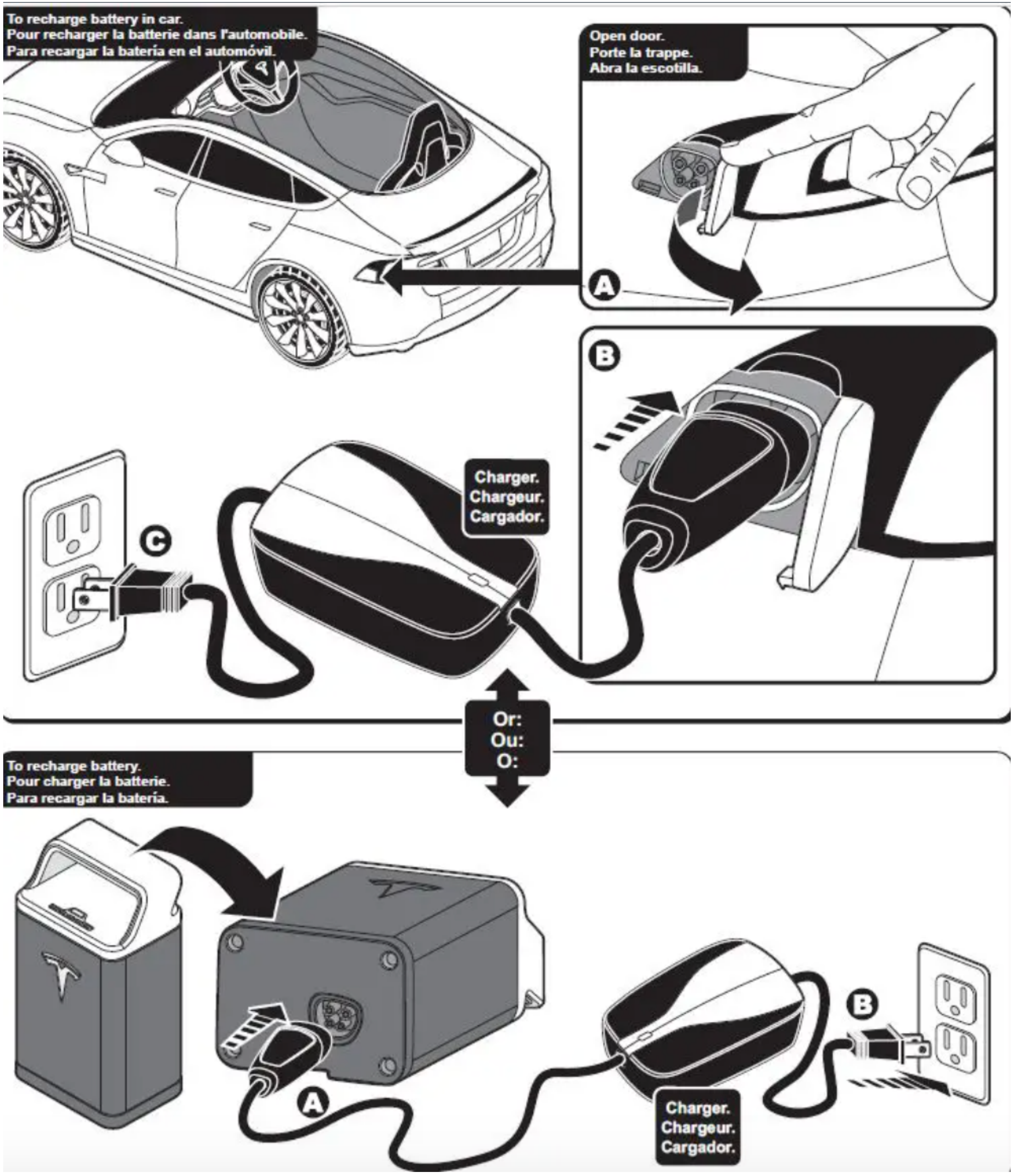

To recharge battery in car.
Pour recharger la batterie dans l'automobile.
Para recargar la batería en el automóvil.
Open door.
Porte la trappe.
Abra la escotilla.
A
B
C
Charger.
Chargeur.
Cargador.
Or:
Ou:
O:
To recharge battery.
Pour charger la batterie.
Para recargar la batería.
A
B
Charger.
Chargeur.
Cargador.

FIGURE 9: BATTERY CHARGING

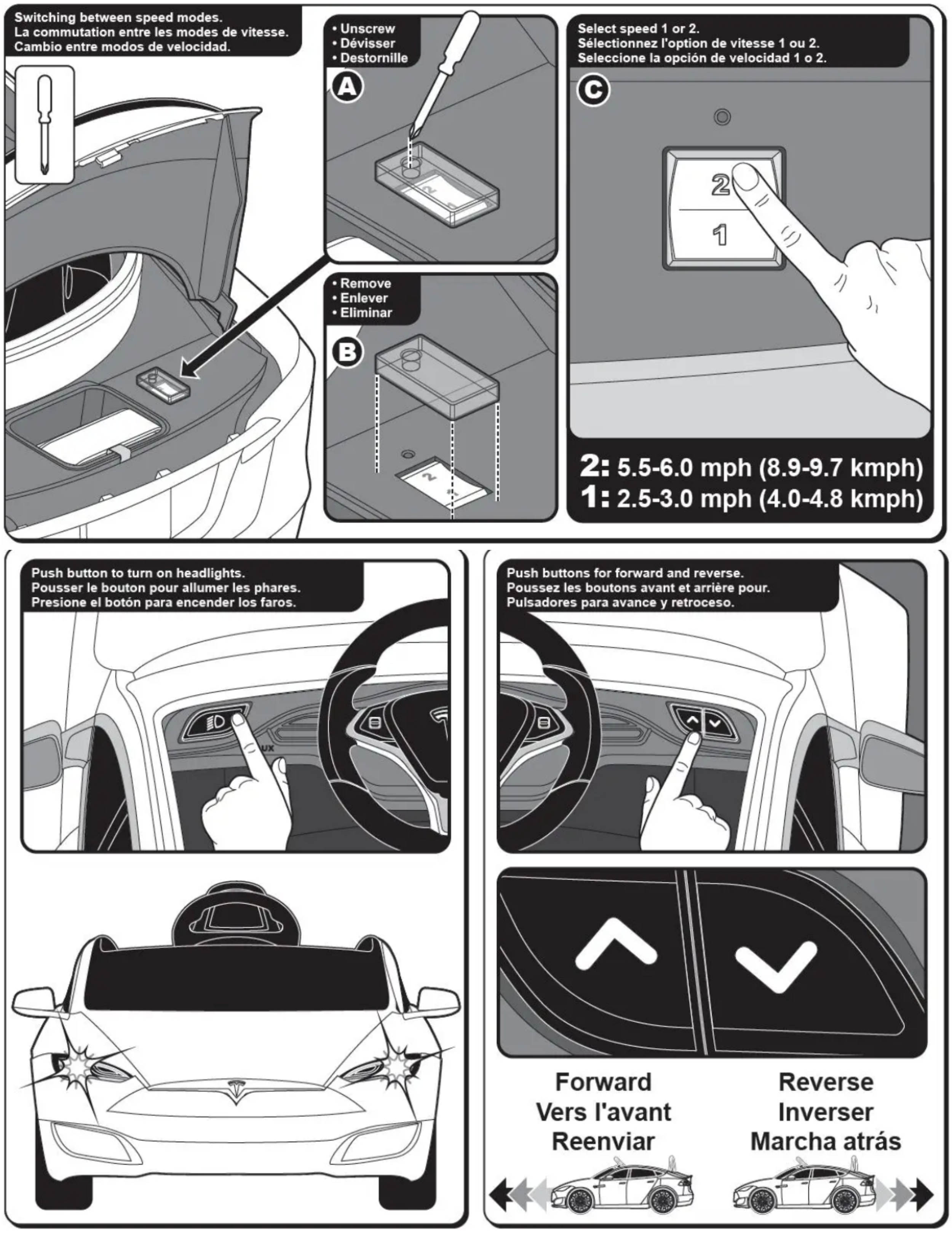

Switching between speed modes.
La commutation entre les modes de vitesse.
Cambio entre modos de velocidad.
• Unscrew
• Dévisser
• Destornille
A
• Remove
• Enlever
• Eliminar
B
Select speed 1 or 2.
Sélectionnez l'option de vitesse 1 ou 2.
Seleccione la opción de velocidad 1 o 2.
C
2
1
2: 5.5-6.0 mph (8.9-9.7 kmph)
1: 2.5-3.0 mph (4.0-4.8 kmph)
Push button to turn on headlights.
Pousser le bouton pour allumer les phares.
Presione el botón para encender los faros.
Push buttons for forward and reverse.
Poussez les boutons avant et arrière pour.
Pulsadores para avance y retroceso.
Forward
Vers l'avant
Reenviar
Reverse
Inverser
Marcha atrás

FIGURE 10: STARTING PROCEDURE

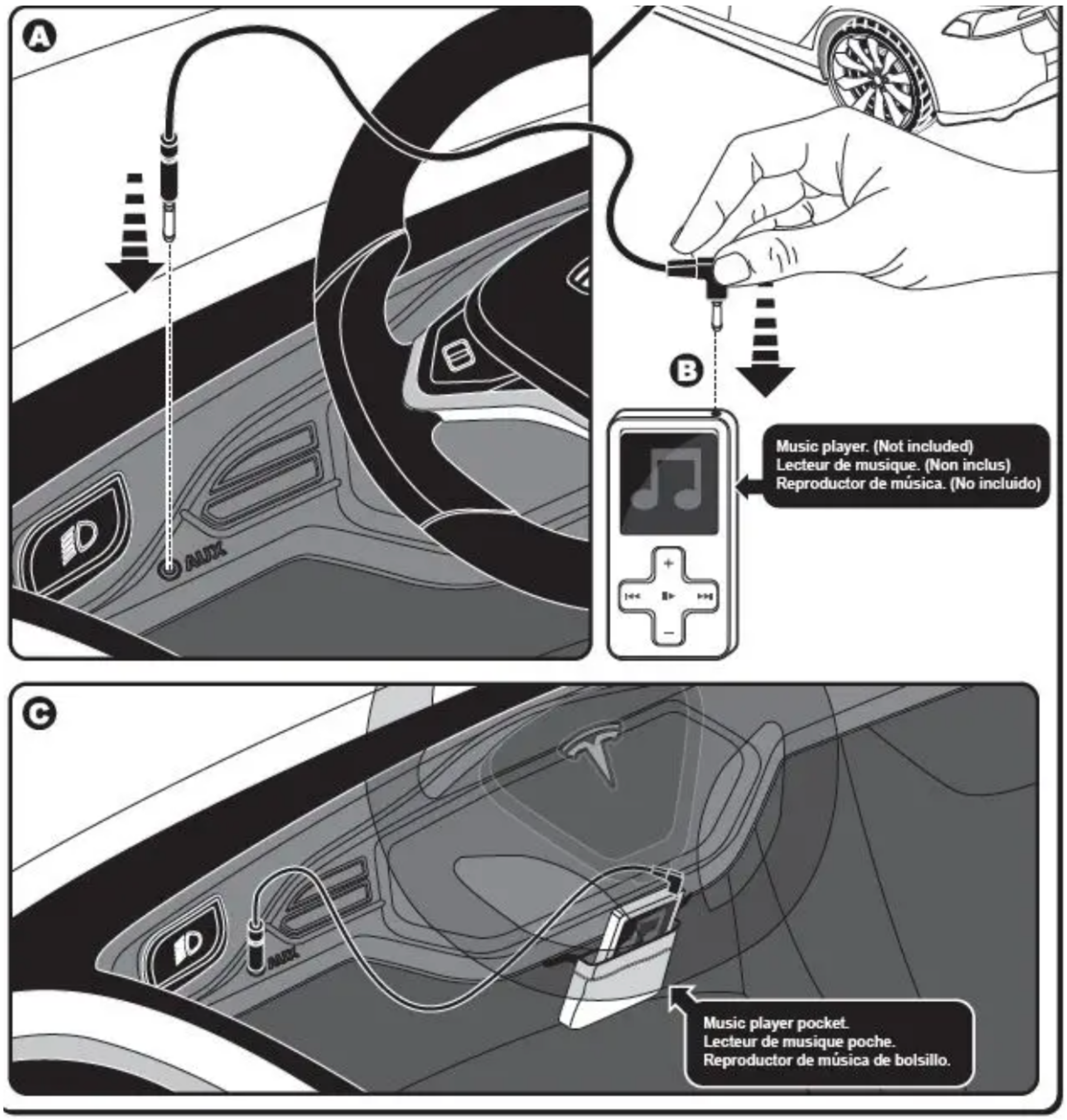

A
AUX
B
Music player. (Not included)
Lecteur de musique. (Non inclus)
Reproductor de música. (No incluido)
C
AUX
Music player pocket.
Lecteur de musique poche.
Reproductor de música de bolsillo.

FIGURE 11: MUSIC PLAYER

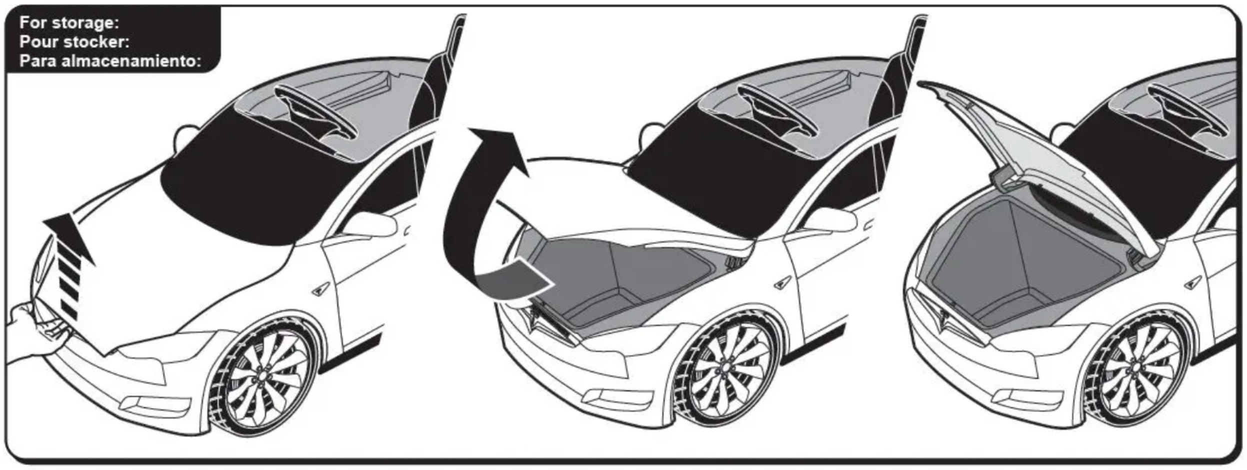


FIGURE 12: STORAGE

Appendix C: Pictures

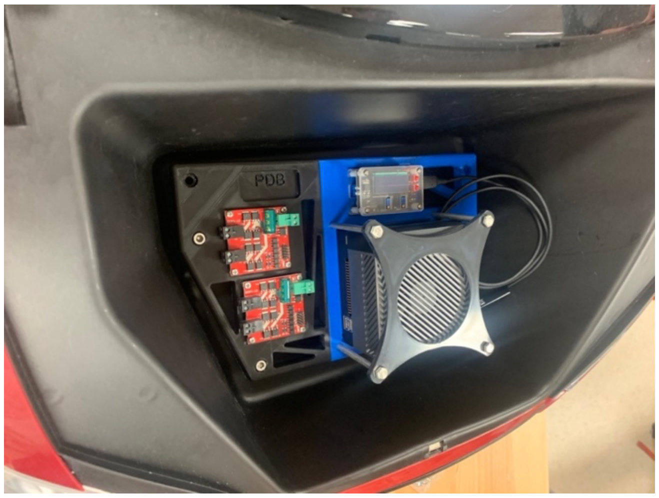


FIGURE 13: NVIDIA JETSON IN FRUNK

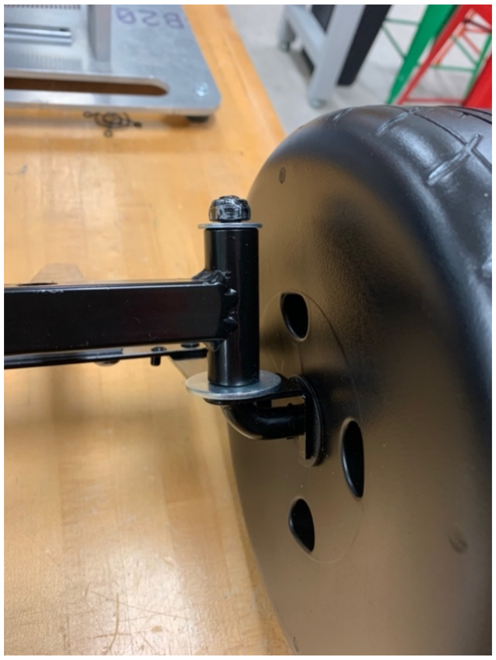


FIGURE 14: FRONT TIRE MECHANICAL CONNECTION

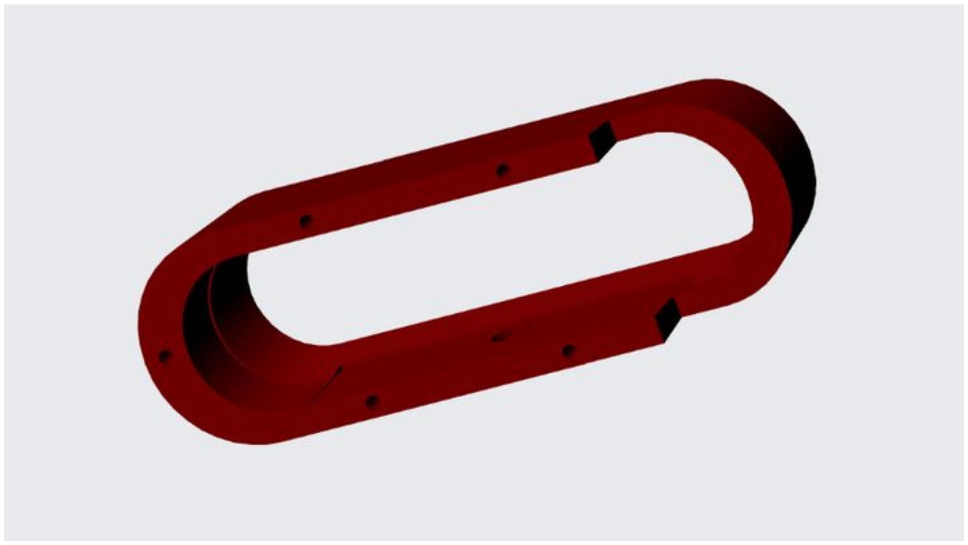


FIGURE 15: CAMERA DISCRETION INITIAL DESIGN

Appendix D: First Code Prototype

```
//TIMER INITIALIZATION
unsigned long ticks = 0;

//function prototypes
void pwm_init(void);
void pwm_set_duty(int channel, int duty);
void command_motor(int channel, int duty);
void timer_init(void);
void Motor_Control(void);

/****************************
* *ENCODER INITIALIZATION* *
****************************/
//Encoder Functions
void encoder1CHB(void);
void encoder1CHA(void);
void encoder_read(void);
void encoder_init(void);

//pins for encoder 1
const byte interruptPIN2 = 2;
const byte interruptPIN3 = 3;

//encoder Variables
byte state1 = 0;
byte state2 = 0;

//current value at pulse
long int encoder_val = 0;

/******************************
* *CONTROLLER INITIALIZATION* *
******************************/

//Controller Gains
#define Kp 20
#define Kd 10
#define Vcc 7.2
int duty_cycle = 0;

//Position variables
float Motor_Angle = 0.0;
float des_Motor_Angle = 0.0;
float prev_des_Motor_Angle = 0.0;
float Act_Motor_Angle = 0.0;
float Motor_Ang_Velocity = 0.0;
float des_Motor_Ang_Velocity = 0.0;
float des_Voltage = 0.0;

//Constants
#define pi 3.1415926535
#define Gear_Ratio 165
#define dT 0.01
#define Arm_Length 0.5
#define PPL 12

void setup() {
  // put your setup code here, to run once:
        timer_init();
        encoder_init();
        pwm_init();

        Serial.begin(9600);
}

void loop() {
        Serial.print("Variable_1:");
        Serial.print(des_Motor_Angle);
        Serial.print(",");
        Serial.print("Variable_2:");
        Serial.println(Act_Motor_Angle);
}

ISR(TIMER1_COMPA_vect){
        Control();
}

/****************************************************
* ************** USER CONTROL ********************** *
****************************************************/

void Control(){
        if (ticks <= 300){
                des_Motor_Angle = 0.0; //PUT A CALIBRATION HERE
        }
        else if (ticks < 2500){
                des_Motor_Angle = 2 * pi * sin(pi * ticks / 1000); //READ IN DATA TO
FOLLOW HERE
        }
        else{
                des_Motor_Angle = 0;
		}


	  /********** GIVEN DONT CHANGE ********/
		Get_Current_Pos();
		Motor_Control();
		ticks++;
	}

	/****************************************************
	* ************* ENCODER SET UP ******************* *
	****************************************************/

	void encoder1CHA(void)
	{

		//digitalReadFast(interruptPIN2);
		if ((state1 == 0) && (digitalRead(interruptPIN2) == 1)){
		//if ((state1 == 0) && (digitalReadFast(interruptPIN2) ==1)){
			state1 = 2;
			encoder_val -= 1;
		}
		else if ((state1 == 2) && (digitalRead(interruptPIN2) == 0)){
		//else if ((state1==2) && (digitalReadFast(interruptPIN2) ==0)){
			state1 = 0;
			encoder_val += 1;
		}
		else if ((state1 == 1) && (digitalRead(interruptPIN2) == 1)){
		//else if ((state1==1) && (digitalReadFast(interruptPIN2) ==1)){
			state1 = 3;
			encoder_val += 1;
		}
		else if ((state1 == 3) && (digitalRead(interruptPIN2) == 0)){
		//else if ((state1==3) && (digitalReadFast(interruptPIN2) ==0)){
			state1 = 1;
			encoder_val -= 1;
		}
	}

	void encoder1CHB(void){
		if ((state1 == 0) && (digitalRead(interruptPIN3) == 1)){
		//if ((state1 == 0) && (digitalReadFast(interruptPIN3) ==1)){
			state1 = 1;
			encoder_val += 1;
		}
		else if ((state1 == 3) && (digitalRead(interruptPIN3) == 0)){
		//else if ((state1==3) && (digitalReadFast(interruptPIN3) ==0)){
```

```
                state1 = 2;
                encoder_val += 1;
            }
            else if ((state1 == 2) && (digitalRead(interruptPIN3) == 1)){
            // else if ((state1==2) && (digitalReadFast(interruptPIN3) ==1)){
                state1 = 3;
                encoder_val -= 1;
            }
            else if ((state1 == 1) && (digitalRead(interruptPIN3) == 0)){
            //else if ((state1==1) && (digitalReadFast(interruptPIN3) ==0)){
                state1 = 0;
                encoder_val -= 1;
            }
        }

        /****************************************************
        * ************** CONTROLLER *************** *
        ****************************************************/

        void Get_Current_Pos(void){
            //Set up for prev pos
            static float Prev_Pos = 0.0;
            Act_Motor_Angle = encoder_val * 2 * pi / (4 * 1980);
            Motor_Ang_Velocity = (Act_Motor_Angle - Prev_Pos) / dT;
            Prev_Pos = Act_Motor_Angle;
        }

        void invKinematics(void){
            des_Motor_Ang_Velocity = (des_Motor_Angle - prev_des_Motor_Angle) / dT;
        }

        void Motor_Control(void){
            int i;
            invKinematics();

            des_Motor_Angle = prev_des_Motor_Angle + des_Motor_Ang_Velocity * dT;
            prev_des_Motor_Angle = des_Motor_Angle;
            des_Voltage = Kp * (des_Motor_Angle - Act_Motor_Angle) + Kd *
(des_Motor_Ang_Velocity - Motor_Ang_Velocity);

            duty_cycle = des_Voltage / Vcc * 255;

            if (duty_cycle > 255)
                duty_cycle = 255;
            else if (duty_cycle < -255)
                duty_cycle = -255;
            command_motor(0, duty_cycle);
        }

        /****************************************************
        * ************* MOTOR SET UP *************** *
        ****************************************************/

        void command_motor(int channel, int duty){
            if (channel == 0){
                if (duty > 0) PORTB = (PORTB & 0b11111101) | 1;
                else PORTB = (PORTB & 0b11111110) | 2;
            }
            else{
                if (duty > 0) PORTB = (PORTB & 0b11110111) | 4;
                else PORTB = (PORTB & 0b11111011) | 8;
            }
            pwm_set_duty(channel, abs(duty));
        }

        void pwm_set_duty(int channel, int duty){
            if (channel == 0)
                OCR5A = duty;
            else if (channel == 1)
                OCR5B = duty;
        }


        /****************************************************
        * ************* ALL INITIALIZATIONS *************** *
        ****************************************************/

        void pwm_init(void) {
            pinMode(45, OUTPUT);
            pinMode(46, OUTPUT);

            TCCR5A = _BV(COM5A1) | _BV(COM5B1) | _BV(WGM52) | _BV(WGM50);
            TCCR5B = _BV(CS51) | _BV(CS50);  //set prescaler to 128
            OCR5A  =  0;
            OCR5B  =  0;
        }

        void timer_init(void){
            // initialize timer1
            noInterrupts();          // disable all interrupts
            TCCR1A = 0;
            TCCR1B = 0;
            TCNT1  = 0;
            //625 for 100Hz, 125 for 500Hz
            OCR1A = 625;            // compare match register 16MHz/256/2Hz
            TCCR1B |= (1 << WGM12);   // CTC mode
            TCCR1B |= (1 << CS12);    // 256 prescaler
            TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt
            interrupts();           // enable all interrupts
        }

        void encoder_init( void){

            // encoder
            pinMode(interruptPIN2, INPUT_PULLUP);
            pinMode(interruptPIN3, INPUT_PULLUP);
            attachInterrupt(digitalPinToInterrupt(interruptPIN2), encoder1CHA, CHANGE);
            attachInterrupt(digitalPinToInterrupt(interruptPIN3), encoder1CHB, CHANGE);
        }
```