
# Trackless Tram System

Supervisor: Dr. Elrabaa

**By:**

**Ayman Kurdi**

**Basil Hashim**

**Faisal ALkharboush**

Senior Project

# Contents

# 1 Introduction:

Most big cities use fossil fueled buses for public transportation for its residents. These buses produce large quantities of heat which contribute 18% of total emissions [ucsusa.com]. The emissions and the heat dramatically reduce the quality of life of both citizens and visitors of the town. However it is not an easy task to switch to tram systems due to the unrealistically large initial capital investment required. According to tahoetram.com, the required initial investment can reach almost 500,000,000 dollars. Most of the initial required capital investment would go towards building the infrastructure. The installation process of these tram systems is also labor intensive as well as intrusive to the lives of the citizens of that particular city.

# 2 Problem Statement

This project describes a trackless (railway-free) multi-car electric tram (dubbed the tram bus) that attempts to solve the problem cost bringing it down to $225M from $500M and reducing emissions by moving away from the dependency on fossil fuel. This tram system should also be able to move freely inside the city.

Public transportation helps in reducing traffic, risk of accidents and emissions. However, having an efficient city wide transportation system that can go through traffic properly is hard to implement and costs tremendous amounts of money. This project aims to solve those problems.

## 2.1 Positive impact.

If successful, this project can provide large incentive to implement public transportation to cities that don't have one in place already. In addition to reducing traffic and emissions, it will also be relatively cheap to implement.

## 2.2 Possible negative impact, due to misuse or unaccounted for ri

Since this system should move freely within the city, it could cause unwanted traffic congestions if the driver moves around lanes too much.

# 3 Project specifications

## 3.1 Requirements

According to proposed solution provided by Dr.Elrabaa', we are required to build a small concept of the original trackless tram described in the introduction, following is the requirements that should be considered for the project:

- A Pilot (master) car controlled by the user.
- Cars attached to the pilot car (slaves) and follow its path precisely, without any control by the user.
- Capable to do sharp turns.
- The pilot must share his speed and direction with his slaves.
- Extendibility of the number the attached cars.
- Make them start moving and stopping at the same time.
- The pilot must have a break system.
- The cars are equipped with normal bus-like rubber tires.
- The controllers of slaves' cars must control the amount, position, and timing of steering of each car such that all cars follow in the exact path that was traveled by the pilot.
- There must be a fixed distance between the cars, so they would not crash with each other.

## 3.2 Technical specifications:

The system consists of two types of cars, pilot and trailers cars. There is only one pilot and two trailers cars. The pilot is controlled by the user. Each car is responsible of sending feedback of their track and speed to the next car. Trailer cars should follow the path they receive.

## 3.3 Taken Approach

- All cars are equipped with an Arduino UNO microcontroller responsible for controlling all components of the car.

- Each car is equipped with two DC motors responsible for controlling the speed of the car. The power to those motors is controlled by the microcontroller to define their rotation speed.
- Each car is equipped with a stepper motor that is responsible for the steering angel and is controlled by the microcontroller.
- Each DC motor is equipped with a light sensor to read how many rotation the wheel finished, this is done by counting the white and black areas.


Figure 1: The light sensor

- Each car is equipped with an XBee module connected to the microcontroller to take care of the communication between the cars.
- An XBee connected to a computer to support user input to the pilot car.
- The cars have three wheels two in the back, one in the front.
- We will build a steering mechanism using the Fab-Lab and attached it to the body of car.

Figure 2: the body with rear wheels showing the DC motors and the light sensors

The equipment were selected specifically to meet the technical reequipments of this project. Arduino UNO provides an easy programmable interface. XBee can provide mesh networks and packet or serial communication. DC motor are small light have

high speeds and less power consumption and is ideal for speed control. Stepper motors can turn in accurate angels and is ideal for steering the car.

# 4 Task Schedule

We divided the project into tasks assigning each of which to who is the most experience and capable of doing that task with ease as a leader to this task. The rest of the team of course support him and work in the subtasks. In this table we listed the tasks, subtasks, the task owner and the duration (when is should start and when it should be finished):

| Task id | Task | Members Assigned | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1.0 | Hardware | A | | x | x | x | x | | | | | | | | | | |
| 1.1 | Deciding | A,B,F | | x | | | | | | | | | | | | | |
| 1.2 | Ordering | F | | | x | x | | | | | | | | | | | |
| 1.3 | Hardware setup | A | | | | | x | | | | | | | | | | |
| 2.0 | Car Modeling | B | | | | x | x | x | x | x | x | | | | | | |
| 2.1 | Body Shape | F,B | | | | x | x | x | | | | | | | | | |
| 2.2 | Steering Mechanism | B | | | | | | | x | x | | | | | | | |
| 2.3 | Fabricating the designed models | B | | | | | | | x | x | x | | | | | | |
| 3.0 | Software | B | | | | x | x | x | x | x | x | x | x | | | | |
| 3.1 | Speed and angel control system | A,B,F | | | | | | x | x | x | x | x | x | | | | |
| 3.2 | Feedback system | B,F | | | | | x | x | x | | | | | | | | |
| 3.3 | Communication system | F | | | | | x | x | | | | | | | | | |
| 4.0 | Communication setup | F | | | | | x | x | | | | | | | | | |
| 5.0 | Finalizing project | B | | | | | | | | | | x | x | x | x | x | x |
| 5.1 | Testing | A,B | | | | | | | | | | x | x | x | x | | |
| 5.2 | Debugging | A,B,F | | | | | | | | | | | x | x | x | | |
| 5.3 | Demo | A,B,F | | | | | | | | | | | | | | x | x |
| 6.0 | Report | A,B | | x | x | x | x | x | x | x | x | x | x | x | x | x | x |
| 6.1 | Documenting | A,B,F | | x | x | x | x | x | x | x | x | x | x | x | x | | |
| 6.2 | Presentation file | A,B,F | | | | | | | x | x | | | | | x | x | x |

Table 1: The tasks list in a time span

- Note: We discuss the plane again and we decided to add the car modeling to it.

The description of the listed tasks is as follows:

| Task | Description |
|---|---|
| Hardware | The team decide the parts (microcontrollers, feedback sensors and others) needed and what's the best to buy. Then, one of the teams gets us the parts to setup the circuit. |
| Car Mechanics | Where we select the proper motors based on the torque, size and accuracy. Also in this part, we design the steering mechanisms and form the shape taking into account the size and beauty. |
| Software | We chose the platform to program our system on and we start programming the cars. |
| Communication | We select a protocol to communicate between the pilot car and trailing cars. And configure the transmitter(s) and receiver(s) |
| Finalizing the project | We test and debug the whole project and see if there is something that needs to be fixed before we get the demo ready. |
| Report | Every single member is responsible to document his own work and submit it to this (Report) task owner. |

# 5 Completed Tasks

We've done a lot so far and yet there is a lot to do. Some tasks were easy to do and doing them was just a matter of time, whereas others were quite difficult but still doable we are going to describe that with each completed task:

## 5.1 Choosing the equipment and purchasing them:

We agreed to meet on a time after each of us read about RC cars motors, microcontrollers and sensors the member thinks the team might needs. We met and discuss what the best is. We finalize a purchasing list, showed it to the supervisor and we order them. But we suffer from the suppliers because their prices are more expensive in comparison with the prices abroad, so we had to get our things from there which took more time than it should.

## 5.2 Hardware Implementation:

It was delayed because we got the components late. Once we received the parts we set everything up.

## 5.3 Car modeling:

After we got the car body we started to think about steering using a single wheel because with that body, showing in Figure-3, it'll be easier to build and also requires less resources as it'll be the same mechanism as the two wheels.

Figure 3: The car body

We knew that we are going to fabricate our own parts so we went to Dhahran fab-lab.


Figure 4: Dhahran Fab-Lab

In this fab-lab and in order to be able to use the machines you have to take the related workshop. The fabrication job could be done by one parson so we assigned this to Basil as he already know about the fab-lab and attended some workshops.

Figure 5: The 3d printer workshop

To try the things we can do in the fab-lab we printed a front wheel using the 3D printer.

Figure 6: 3D printer prototype

Then, we stated thinking about the steering mechanism. We had two choices we either do the differential drive mechanism or we build Ackermann steering geometry[1] but with one wheel. We take the required measurements with help from a mechanical student department and to build rack and pinion gears.

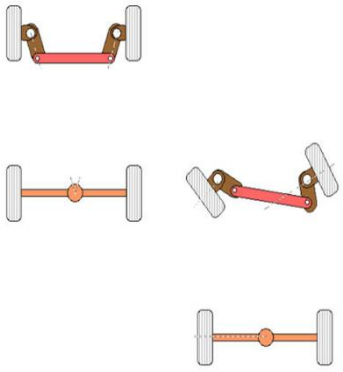

Figure 7: Ackermann steering geometry

We calculated the required lengths for the arms, gear diameter and number of teeth based on some trigonometric equations. The calculations are as follows:

As the maximum degree of turning right/left is 70° we need to determine the shaft length ($2\boldsymbol{x}$) which also give the diameter of the pinion gear and the also the rack. This all could be determined by $\boldsymbol{x} = \boldsymbol{z}\sin 70°$ given $\boldsymbol{z}$, where $\boldsymbol{z}$ is the wheel radius.

- We have $\boldsymbol{z}$ as 34 $mm$; because of car height
- $\boldsymbol{x} = 34\sin 70° = 15.98 \approx 16\ mm$
- The shaft, pinion diameter and rack length:

$$2\boldsymbol{x} = 32\ mm$$

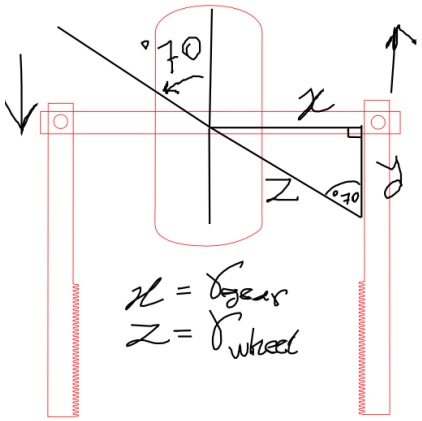


Figure 8: Calculation Description

For the rack and pinion gears design the only thing remaining for the car is fabricating the design, in Figure-8, and but the equipment together.

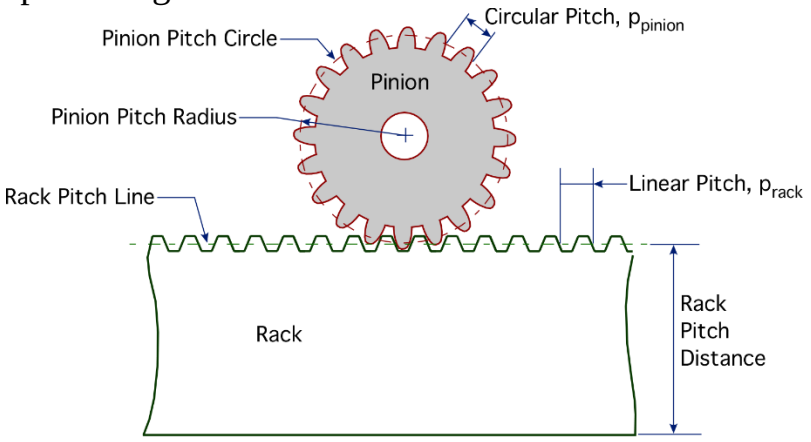


Figure 9: Rack and Pinion Mechanism

[1]: Is a geometric arrangement of linkages in the steering of a car or other vehicle designed to solve the problem of wheels on the inside and outside of a turn needing to trace out circles of different radius [Wikipedia].

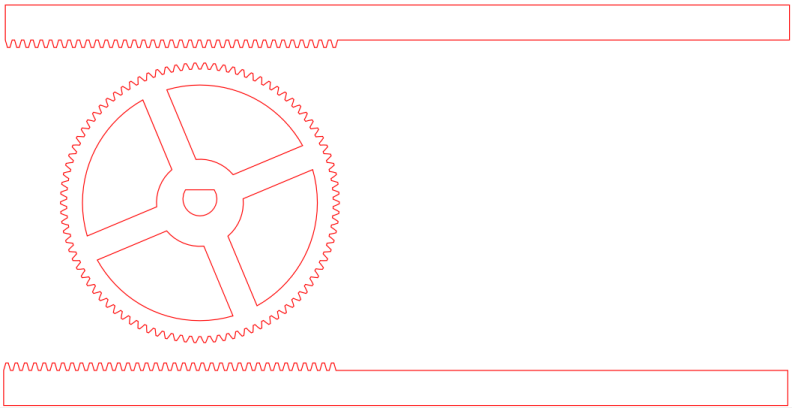

Figure 10: The design we build

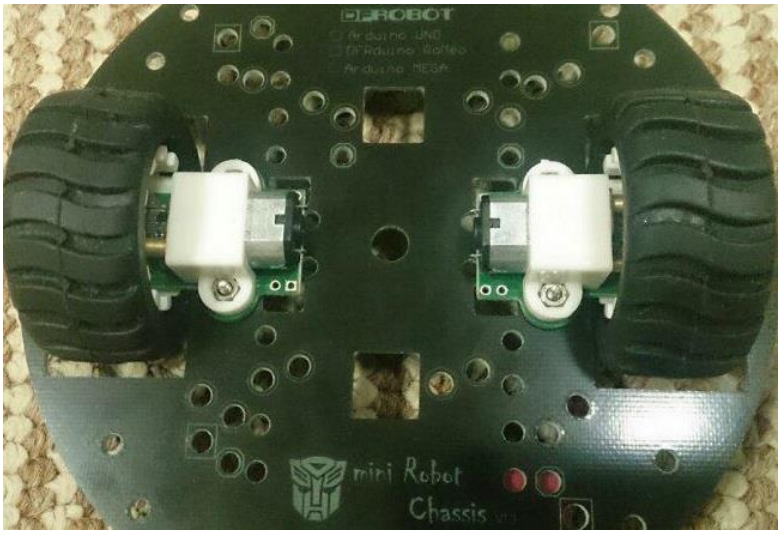


Figure 11: The car body having the DC motors and the feedback sensors installed

## 5.4 Communication:

We have finished setting up and testing the XBEE network. Using mesh network were a coordinator is connected to the computer and each car have its XBee configured as routers. Each router can send directed messages or broadcasts.

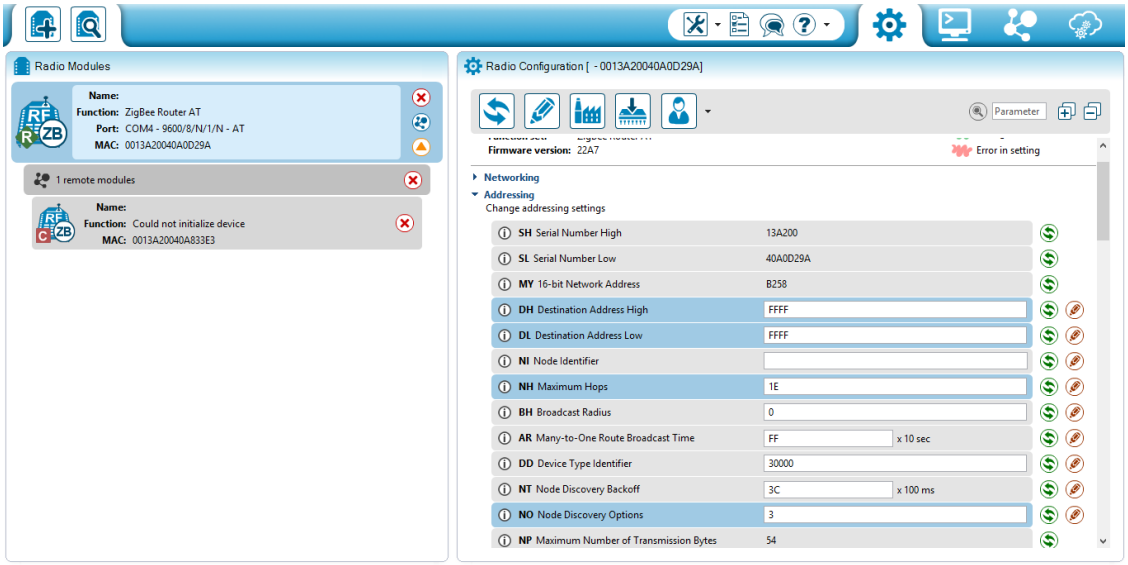


Figure 12: XCTU Software used to configure communication

As showing in Figure-12 one of the router XBee connected to the XBee network coordinator, some of the settings are shown too.

The software for the Arduinos to communicate is finished and is ready to be used (Appendix).

### 5.5 Feedback system:

The code to read from the sensors and calculate the speed is finished and ready to be uploaded and used by the Arduinos (Appendix).

# 6 Conclusion:

This paper has described and presented a solution for cheaper and cleaner tram systems. It has also talked about the goals we set and their feasibility. This paper has described the decision making process and the rationale behind every major decision that has been taken. And concluded by discussing the finished tasks and how the team had proceeded to tackle each obstacle by divided the tasks into more manageable smaller tasks.

Overall the team is happy with the progress that it has made and believes that if it continues with the same pace, we are sure to finish on time.

# 7 Appendix:

## 7.1 Communication

```
/*
This is Xbee Receiver
*/

#include <XBee.h> //include Header File

//#define MSG_LEN 15 //Message Length
XBee xbee = XBee();
ZBRxResponse rx = ZBRxResponse();

int led = 13;
void setup()
{
  xbee.setSerial(Serial);
  Serial.begin(9600);
// pinMode(led, OUTPUT);
  Serial.println("Simple Xbee Communication - Receiver");
}

void loop()
{
xbee.readPacket();  //Reads all available serial bytes until a packet is parsed, an error occurs, or the buffer is empty.
  if (xbee.getResponse().isAvailable()) {  //something available?
      if (xbee.getResponse().getApiId() == ZB_RX_RESPONSE) {  //is it rx packet?

        // now get the payload
        xbee.getResponse().getZBRxResponse(rx);

        Serial.print("Receive packet from (64-bits address) -> ");  //type: XBeeAddress64
        Serial.print(rx.getRemoteAddress64().getMsb(), HEX);
        Serial.print(":");
        Serial.println(rx.getRemoteAddress64().getLsb(), HEX);
```

```
        Serial.print("Packet length is -> ");
        Serial.println(rx.getDataLength(), HEX); //getDataLength() type:
uint8_t, Returns the length of the payload.

        //rx.getData();
        //print the payload
        Serial.println("The packet ...");
        Serial.print("<< ");
        for (byte i=0;i<rx.getDataLength();i++)
          Serial.print(char(rx.getData(i)));
        Serial.println(" >>");
      }
  }

}
```

```
/*
This is Xbee Transmitter
*/

#include <XBee.h> //include Header File
#include <Time.h>

#define MSG_LEN 18 //Message Length
XBee xbee = XBee();// Make a Xbee Variable
uint8_t payload[MSG_LEN];
//XBeeAddress64 addr64 = XBeeAddress64(0x0013A200,0x409022C7);
//define 64-Bit Xbee Adress of the Remote Host
XBeeAddress64 addr64 = XBeeAddress64(0x00000000,0x0000FFFF);
//XBeeAddress64 addr64 = XBeeAddress64(0x0013A200,0x408CC72B);
ZBTxRequest zbTx = ZBTxRequest(addr64, payload, sizeof(payload)); //Make
a Xbee Packet
int led = 13;
void setup()
{
   xbee.setSerial(Serial);
```

```
  Serial.begin(9600);
  //pinMode(led, OUTPUT);
  Serial.println("Simple Xbee Communication - Transmitter");
  setTime(12,10,00,4,3,2013);
  char tmp[] = "COE DEPARTMENT";
  for (byte i=0;i<13;i++)
    payload[i]=tmp[i];

}

void loop()
{
  xbee.send(zbTx);
    //flash led
//  digitalWrite(led, HIGH);
  delay(1000);
//  digitalWrite(led, LOW);
//  delay(3000);
}
```