

## Cargo Solutions

By Ather Hassan, Christopher Vishnu, Ryan Woodard, Mostafa Mohsen

<b>1 Class Index</b>	<b>1</b>
<b>1 Class Index</b>	<b>1</b>
1.1 Class List . . . . .	1
<b>2 File Index</b>	<b>1</b>
2.1 File List . . . . .	1
<b>3 Class Documentation</b>	<b>2</b>
3.1 ControlUnit.gps Class Reference . . . . .	2
3.1.1 Detailed Description . . . . .	2
3.1.2 Member Function Documentation . . . . .	2
3.2 Route.route Class Reference . . . . .	4
3.2.1 Detailed Description . . . . .	4
3.2.2 Member Function Documentation . . . . .	5
<b>4 File Documentation</b>	<b>5</b>
4.1 ControlUnit.py File Reference . . . . .	5
4.1.1 Detailed Description . . . . .	6
4.2 dangerousArea.py File Reference . . . . .	6
4.2.1 Detailed Description . . . . .	6
4.2.2 Function Documentation . . . . .	6
4.3 Route.py File Reference . . . . .	7
4.3.1 Detailed Description . . . . .	8
<b>Index</b>	<b>9</b>

## 1 Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<b>ControlUnit.gps</b>	
Runs all the checks that are needed to ensure that a truck is on the correct route and only operating during the correct times	<b>2</b>
<b>Route.route</b>	
Each route is an instance of this class	<b>4</b>

## 2 File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

**ControlUnit.py**

This unit controls all functions to ensure safety of the vehicles

5

**dangerousArea.py**

A file that checks if a truck is in

6

**Route.py**

This module Consists of the route class which is used to hold a route, improve knowledge about the route, and get data from the routes

7

## 3 Class Documentation

### 3.1 ControlUnit.gps Class Reference

Runs all the checks that are needed to ensure that a truck is on the correct route and only operating during the correct times.

#### Public Member Functions

- `def __init__ (self)`
- `def coordinatesRecieved (self, long, lat, punchedOut)`  
*Runs when any gps coordinates are sent out from the truck.*
- `def OnTrack (self, coors, long, lat)`  
*Calculates if the vehicle has strayed far off the average route, and gives appropriate alerts if it has.*
- `def triggered (long, lat, lastLong, lastLat)`
- `def distance (dx, dy, r)`  
*returns if a point is within r distance away from another*
- `def triggerAccelerometer (self, long, lat)`
- `def setRouteNumber (self, num)`

#### Public Attributes

- `route`
- `lastLong`
- `lastLat`
- `routeNumber`
- `routeNumber`

#### 3.1.1 Detailed Description

Runs all the checks that are needed to ensure that a truck is on the correct route and only operating during the correct times.

If a truck is moving during incorrect hours, without a driver, or too far out of the route it alerts an operator/ the police. Also warns the driver to avoid stops if they are entering a high risk area.

#### 3.1.2 Member Function Documentation

### 3.1.2.1 coordinatesRecieved()

```
def ControlUnit.gps.coordinatesRecieved (
    self,
    long,
    lat,
    punchedOut )
```

Runs when any gps coordinates are sent out from the truck.

Checks that the truck should be operating (it's not a weekend), that the driver is in the truck, and that the latest coordinates are on the path the truck should be on. If any of these checks fail, it sends an appropriate alert.

#### Parameters

<i>long</i>	The incoming longitude
<i>lat</i>	The incoming latitude
<i>punchedOut</i>	Check for if the driver is currently on break

### 3.1.2.2 distance()

```
def ControlUnit.gps.distance (
    dx,
    dy,
    r )
```

returns if a point is within r distance away from another

#### Parameters

<i>dx</i>	difference in horizontal distance
<i>dy</i>	difference in vertical distance
<i>r</i>	the maximum distance

#### Returns

bool returns True if the point is within the distance False if not

### 3.1.2.3 OnTrack()

```
def ControlUnit.gps.OnTrack (
    self,
    coors,
    long,
    lat )
```

Calculates if the vehicle has strayed far off the average route, and gives appropriate alerts if it has.

**Parameters**

<i>routeNumber</i>	The number identifying which route the truck is on
--------------------	--

**Returns**

status The output of whether or not the truck has passed the comparison with average and is currently on route

The documentation for this class was generated from the following file:

- [ControlUnit.py](#)

## 3.2 Route.route Class Reference

each route is an instance of this class

**Public Member Functions**

- def `__init__` (self, DeviceSerial, lat, long, weight, time, routnum)  
*this is the initializer method*
- def `getdata` (self)  
*this method is used to get the longitude and latitude pooints of a route*
- def `avg` (self, fname, recentRout)  
*this method updates a route*
- def `write` (self, fname, lat, long, weight)  
*this method is called to write the route back into the text file which stores it, once it's been updated*

**Public Attributes**

- **id**
- **lat**
- **long**
- **weight**
- **routnum**

### 3.2.1 Detailed Description

each route is an instance of this class

**Parameters**

<i>DeviceSerial</i>	This is the parameter that represents each truck
<i>lat</i>	This variable is the list of latitude values of each point on the route
<i>long</i>	This variable is the list of longitude values of each point on the route
<i>weight</i>	This variable holds the number of routes that have been averaged together so far
<i>lat</i>	This variable is the list of longitude values of each point on the route
<i>routnum</i>	This variable indicates which route is to be created as an object and updated

### 3.2.2 Member Function Documentation

#### 3.2.2.1 avg()

```
def Route.route.avg (
    self,
    fname,
    recentRout )
```

this method updates a route

the method updates a route every time a route has been driven on

the route is updated by taking an average between the current representation of the route and a recently recorded version

the idea behind this is that the representation of a certain route gets more accurate every time it is driven on

#### 3.2.2.2 getdata()

```
def Route.route.getdata (
    self )
```

this method is used to get the longitude and latitude pooints of a route

#### Returns

(self.lat, self.long) the return is a tuples consisting of the list of lat values and list of long values

The documentation for this class was generated from the following file:

- [Route.py](#)

## 4 File Documentation

### 4.1 ControlUnit.py File Reference

This unit controls all functions to ensure safety of the vehicles.

#### Classes

- class [ControlUnit.gps](#)

*Runs all the checks that are needed to ensure that a truck is on the correct route and only operating during the correct times.*

#### Functions

- def **ControlUnit.distance** (dx, dy, r)

#### 4.1.1 Detailed Description

This unit controls all functions to ensure safety of the vehicles.

##### Author

Ather Hassan, Chris Vishnu, Mostafa Mohsen, Ryan Woodard

##### Date

2019/01/27

## 4.2 dangerousArea.py File Reference

A file that checks if a truck is in.

### Functions

- def [dangerousArea.DangerousAreaCheck](#) (lat, long)  
*checks if coordinates are in any of the dangerous areas*
- def [dangerousArea.distance](#) (dx, dy, r)  
*returns if a point is within r distance away from another*
- def [dangerousArea.addDangerousArea](#) (lat, long, r)  
*adds a dangerous area to the file of dangerous areas*

#### 4.2.1 Detailed Description

A file that checks if a truck is in.

##### Author

Ryan, Ather, Chris and Mostafa

##### Date

2019/01/26

#### 4.2.2 Function Documentation

##### 4.2.2.1 addDangerousArea()

```
def dangerousArea.addDangerousArea (  
    lat,  
    long,  
    r )
```

adds a dangerous area to the file of dangerous areas

**Parameters**

<i>lat</i>	latitude of dangerous area
<i>long</i>	longitude of dangerous area
<i>r</i>	radius of the dangerous area

**4.2.2.2 DangerousAreaCheck()**

```
def dangerousArea.DangerousAreaCheck (
    lat,
    long )
```

checks if coordinates are in any of the dangerous areas

**Parameters**

<i>lat</i>	a float that stores the latitude of the truck
<i>long</i>	a float that stores the longitude of a truck

**Returns**

bool True if truck is in a dangerous location False if it is not

**4.2.2.3 distance()**

```
def dangerousArea.distance (
    dx,
    dy,
    r )
```

returns if a point is within r distance away from another

**Parameters**

<i>dx</i>	difference in horizontal distance
<i>dy</i>	difference in vertical distance
<i>r</i>	the maximum distance

**Returns**

bool returns True if the point is within the distance False if not

**4.3 Route.py File Reference**

This module Consists of the route class which is used to hold a route, improve knowledge about the route, and get data from the routes.



## Classes

- class `Route.route`  
*each route is an instance of this class*

## Functions

- `def Route.main ()`

### 4.3.1 Detailed Description

This module Consists of the route class which is used to hold a route, improve knowledge about the route, and get data from the routes.

## Author

Christopher Vishnu, Mostafa Mohsen, Ather Hassan, Ryan Woodard

## Date

Jan 26 2019

## Index

- addDangerousArea
  - dangerousArea.py, [6](#)
- avg
  - Route.route, [5](#)
- ControlUnit.gps, [2](#)
  - coordinatesRecieved, [2](#)
  - distance, [3](#)
  - OnTrack, [3](#)
- ControlUnit.py, [5](#)
- coordinatesRecieved
  - ControlUnit.gps, [2](#)
- dangerousArea.py, [6](#)
  - addDangerousArea, [6](#)
  - DangerousAreaCheck, [7](#)
  - distance, [7](#)
- DangerousAreaCheck
  - dangerousArea.py, [7](#)
- distance
  - ControlUnit.gps, [3](#)
  - dangerousArea.py, [7](#)
- getdata
  - Route.route, [5](#)
- OnTrack
  - ControlUnit.gps, [3](#)
- Route.py, [7](#)
- Route.route, [4](#)
  - avg, [5](#)
  - getdata, [5](#)