

Brief Description of the Autonomous Golf Cart Simulator

Danko Antolovic, 01/2007

Introduction

CartSim is the simulator for the autonomous golf cart built in the IU's Department of Computer Science. Its purpose is to help in the development of driving programs, and to visualize/verify the route that the real cart will take under the control of such a program.

Like the cart itself, the simulator exchanges messages with a driver program at fixed time intervals, through a network link. The messages from the cart/simulator contain the state of the cart (current steering direction, speed, location), and the messages from the driver should contain driving instructions (e.g. desired direction and speed).

CartSim is a regular graphic application, running under Windows or Linux. The driver program consists of two main units: the communication interface, which keeps track of the synchronous communication with the cart/simulator, and the driving module, which makes all the decisions about driving. This latter module is called up at regular time intervals: when it is called, it must check what the cart is doing, decide what to do next, and tell the cart to do it. Writing the driving module is your main objective, and the simulator demo comes with the completed communication interface and a few simple driving modules for demonstration. Your driving modules can be ported from the simulator to the autonomous cart without modifications.

The graphic window of the simulator represents the driving area. Like the real cart, the simulator initially sits still and waits for a driver to call upon it. The driver establishes connection, and starts sending instructions; the simulator draws the cart's trajectory on the ground, displays some of the cart's parameters, and provides an emergency stop and restart. The driver and the simulator can run on the same machine, or across a network and under different operating systems.

Functional description of the simulator

The simulator can represent the driving area in two ways:

- ad hoc: the driving area is a 50m x 50m square, overlaid with a 1m x 1m grid. This display has no geographic orientation, and is suitable for simulating simple routes, in which there are no pre-set points of call.

- geographic: the driving area encloses a set of waypoints, and is overlaid with a grid of meridians and parallels, at intervals of 0.00001 degrees of longitude and latitude (Figure 1). The waypoints are pre-defined locations which the cart must pass, and they are given as latitudes and longitudes on a geographic map. They are loaded into the simulation from a waypoints file, which is named "RDDF.txt", and must be present in the simulator's directory (see the example file, which is provided with the simulator; also Figure 3). In the real driving exercises, the waypoints will be actual marked locations of the ground, and you will be given the corresponding waypoints file to use in your driver program.

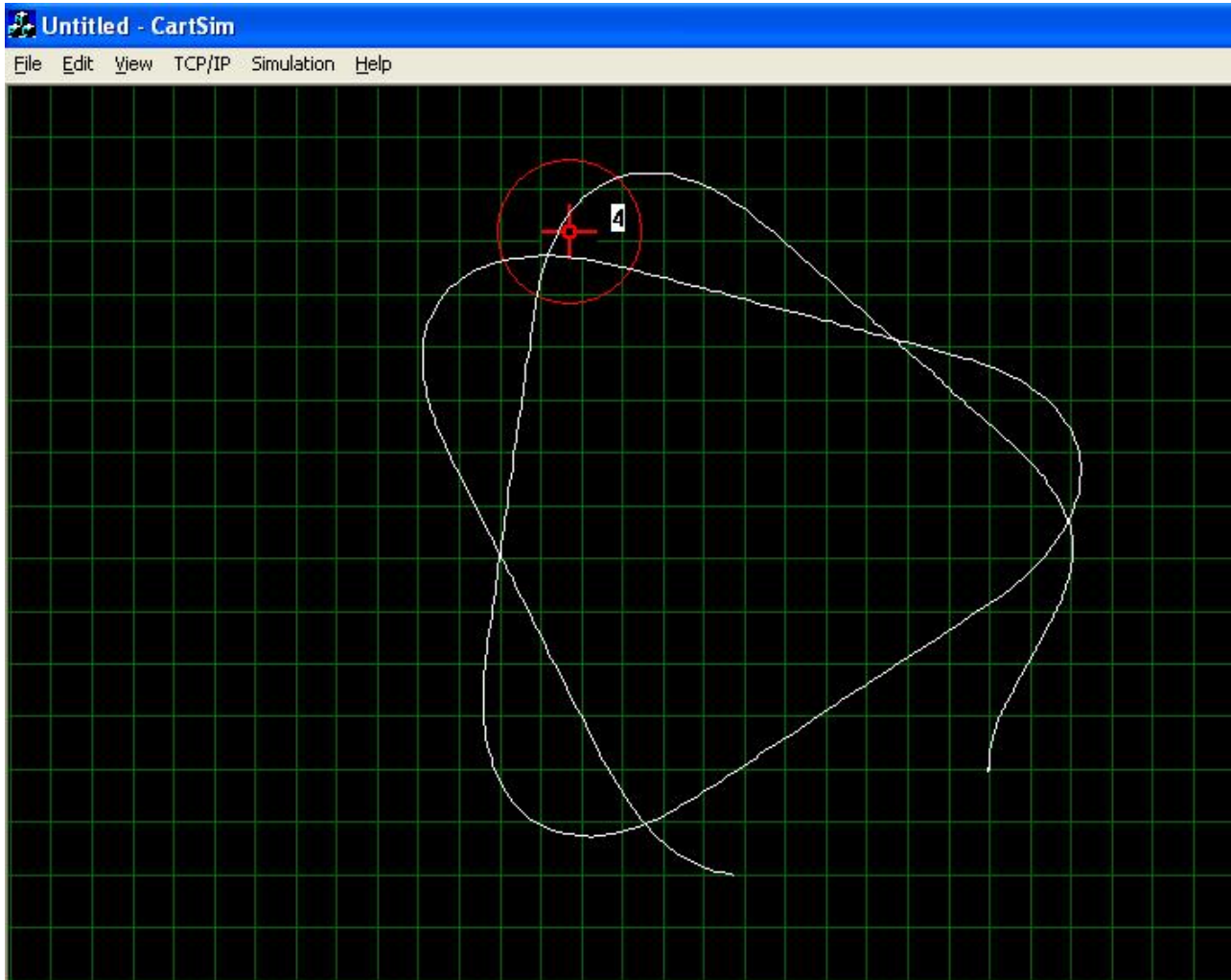


Figure 1: Part of the driving area, in geographic representation. The image shows a short trajectory in white, a waypoint as a red mark, and the grid of meridians and parallels in the background.

The simulator starts up in the ad-hoc display by default. To switch to geographic display, select menu "Simulation/Load waypoints"; to return to the ad-hoc display, select "Simulation/Clear waypoints." In both cases the cart is initially positioned in the center, facing up (north). This initial location can be changed by right-clicking at the desired location in the window.

The simulator reports the cart's current location and velocity vector as simulated GPS data. The GPS data are refreshed at a set frequency, and the reported locations have a Gaussian error, relative to the true location of the cart. Menu "GPS/Options" allows you to choose the frequency (once or five times per second), and to set the standard deviation (σ) of the location error. Setting $\sigma = 0$ provides an error-free location in the simulation, but the cart's real GPS has a fixed σ , which can not be turned off!

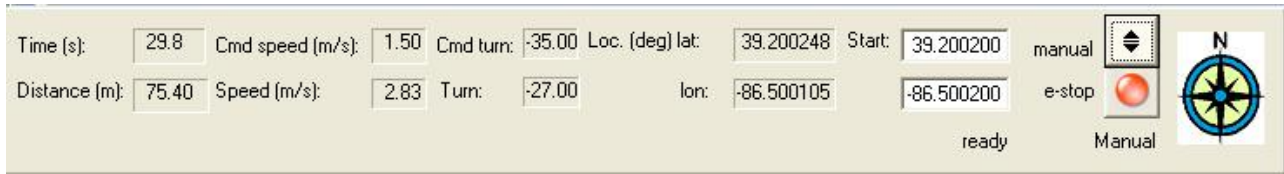


Figure 2: Simulator's control panel. The compass' rose indicates geographic display of the driving area.

A movable panel (normally located at the bottom of the display window, Figure 2), displays simulation controls and a selection of data pertaining to the cart and the trajectory:

- total time and distance of travel,
- commanded and actual control values (speed and turn),
- location of the cursor (either as a distance from the center, in meters, or as latitude and longitude),
- initial location of the cart,
- current status of the network connection between the driver and the cart,
- current mode of the cart (see the section below).

Also located on the panel are the safety switch and the e-stop button (both are implemented as toggle buttons).

Menu command "Simulation/Start" starts the communication server and waits for the driver's call. The default port is 5000, but the port number can be changed with the menu command "TCP/Port".

During the simulation, all items on the "Simulation" menu are inactive, except "Stop".

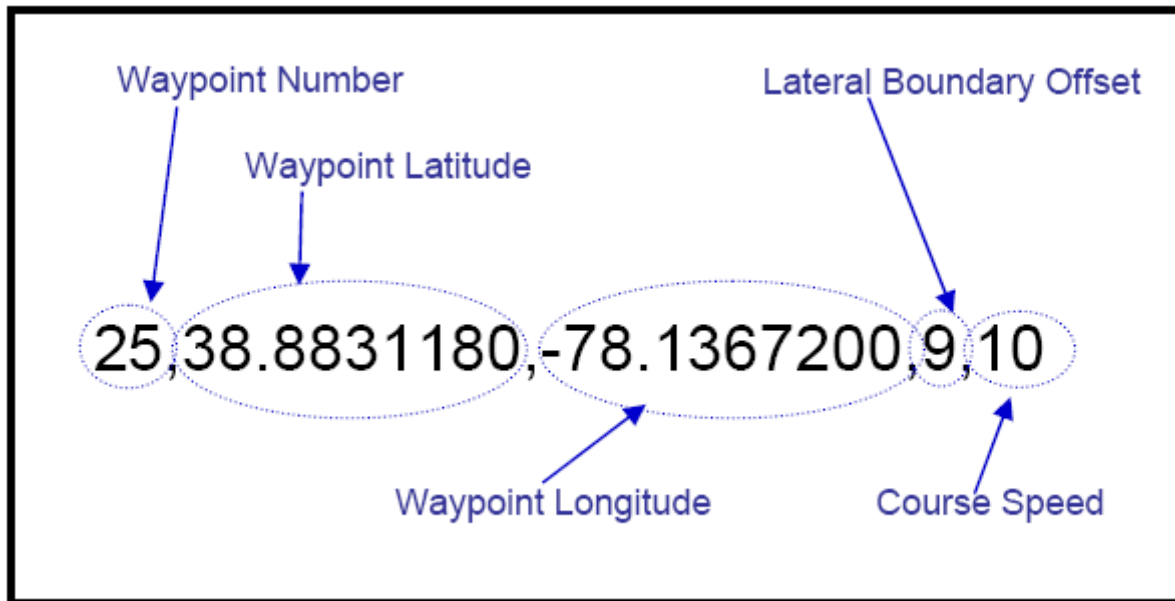


Figure 3: Data describing one waypoint in the RDDF (source: http://www.darpa.mil/grandchallenge05/RDDF_Document.pdf)

The driving modes

The autonomous cart has several driving modes, which correspond to common situations: it can be turned off, starting up, driven by a human (manual), driven by a program (autonomous), stopped in an emergency (e-stop), or in some condition of failure.

The simulator implements the cart's driving modes, as shown in the diagram in Figure 4. For safety reasons, the cart's default mode is manual. The cart obeys the driving program only in its autonomous mode, and it can be placed in that mode only from the manual mode, by the human driver lifting a safety handle. For the cart to transition into autonomous mode, it must also already be in contact (exchanging messages) with a driver program. If the messaging stream stops (times out), the cart reverts to the manual mode.

In the simulation, when you are ready to let your code do the driving, you will place the cart in the autonomous mode by clicking the up-down arrow on the control panel.

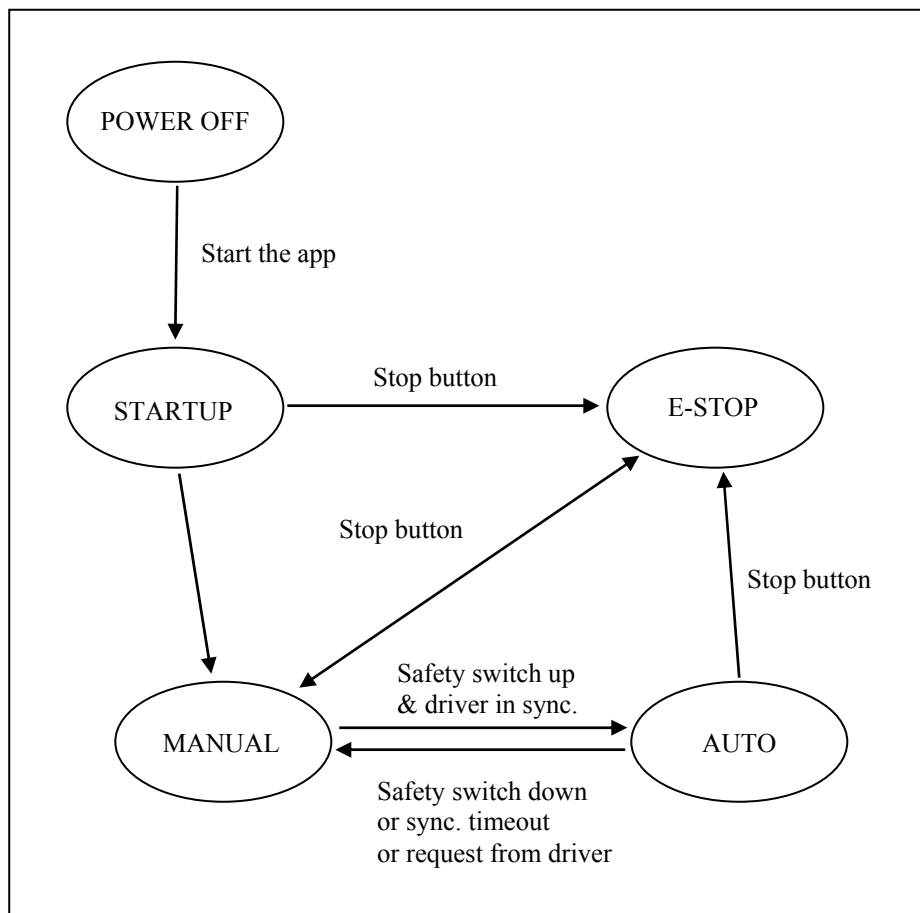


Figure 4: The simulator's driving modes. The physical cart has some additional modes, corresponding to failure conditions. The structure of this diagram is determined to a large extent by safety considerations in operating an autonomous vehicle.

The cart also has an emergency stop button (the red button in the simulator). Pressing that button at any time places the cart in emergency stop: it will not run and will not obey the driving code. Pressing the e-stop again puts the cart into manual mode, from which it can be re-enabled only by throwing the safety handle up. You can simulate these actions in the control panel, but the real cart would only be e-stopped by the human safety driver in the case of emergency or malfunction.

For safety reasons, the driver program can only place the cart into manual mode; it can not reach the other modes. Since the cart (and the simulator) obey the driving code only in the autonomous mode, your driving program should continuously check the mode. If the cart exits the autonomous mode, the driver program should suspend its driving actions until the cart is autonomous again.

The Control Frame

The cart and the driver exchange messages in the form of text strings, and the communication interface updates an identical local data structure in both the cart and the driver, every time it receives a message. That data struct, the Control Frame, reflects the current state of the cart and the driver's intended actions.

The pointer to the Control Frame is passed to the driving module; your driving code will read the cart status and place driving instructions in the appropriate fields (see the sample driving modules in the demo).

```
// structure containing the data that are passed back and forth in the
// control messages
struct ControlFrame
{
    long msgCount;           // count of messages sent by the cart
    long timeStamp;        // cart-maintained clock (ms since start-up)

    float cmdAngle;        // desired wheel angle (degrees from center)
    float curAngle;        // current wheel angle
    char angleLimitErr[10]; // indicator that the angle is at its limit

    float cmdSpeed;        // desired speed (m/s)
    float curSpeed;        // current speed
    float pcntThrottle;    // % of full throttle
    char speedLimitErr[10]; // indicator that the speed is at its limit

    float GPSLat;          // GPS latitude (degrees)
    float GPSLon;          // GPS longitude (degrees)
    float GPSVelEast;     // eastward velocity (m/s)
    float GPSVelNorth;    // northward velocity (m/s)
    float GPSEpe;         // GPS estimated position error (m)
    char GPSStatus[10];   // status/availability of the GPS

    char connStatus[10];   // status of the cart/driver connection
    char cmdSystemMode[10]; // desired cart mode
    char systemMode[10];  // cart's driving mode
    int  errorCode;       // current error number
};
```

Running the simulation demo under Windows

Start the CartSim application, select the display mode, adjust the starting point if desired, then select menu item "Simulation/Start". Run the demo version of the Driver as a command-line app, from the command window. Driver takes two arguments, IP address of the server (cart), and the port number; for the loopback connection and the default port, the command looks like this:

```
C:\DOS> Driver 127.0.0.1 5000
```

Upon starting, the driver program calls up the cart's server, validates itself, then goes on scrolling a summary line for every received message. Remember that the messages must be exchanged as long as the driver has a valid connection with the cart, regardless of the actual driving. The driver can not steer the cart unless the cart is in the automatic mode.

Put the cart in the automatic mode with the safety switch (the up/down arrow), and the CartSim begins to plot the cart's route as a white line. In this demo, the cart runs in rosetta-like patterns, which can be used to assess the reliability of the network connection: if messages are missed or delayed, the pattern becomes distorted.

E-stop button places the cart in the e-stop mode; releasing the button places it into manual mode. The cart can be placed in automatic mode only by throwing the safety switch from manual to automatic.