Overbot Documentation

Autonomous Systems Lab
Summer 2006
Eli Kwitman & Chris Woodruff

Abstract: This document covers the large changes to the Overbot and directions for how things work on the vehicle. An introduction to CVS and documentation on the current Overtux server configuration are also given as a guide for future users.

Table of Contents

1 VEHICLE NETWORK SETTINGS 3
2 HOW TO CREATE A WAYPOINT FILE 3
2.1. Create Waypoint File 3
2.2. Edit the Waypoint File 3
2.3. Move the Waypoint File 4
3 HOW TO RUN THE OVERBOT AUTONOMOUSLY 4
4 MODIFICATIONS MADE TO THE OVERBOT 6
4.1. BLOWER FANS 6
4.2. AUTONOSLUG BOX 6
4.3. AC 120V OUTLET 9
5 CVS server 9
5.1. Password and Writers File 9
5.2. File Permissions 9
5.3. Lock Folder 10
5.4. Normal CVS USAGE 10
6 Miscellaneous **Error! Bookmark not defined.**

1 VEHICLE NETWORK SETTINGS

The laptop can access the internet when the "network settings" have been set to DHCP. Before the vehicle network can be accessed, the settings have to be put into "manual" and the IP of the laptop (Baja) has to be specified (we have been using 10.100.100.128 with a net mask of 255.255.255.0) along with the domain name "overbot.org"

To access the native networking of QNX use the /net directory. All computers connected to the network are listed there.

For example:
files on the Overbot computer can be accessed using the command "ls /net"

2 HOW TO CREATE A WAYPOINT FILE

The laptop baja can access the main vehicle computer "gcrear0" via ethernet cable. There are ports on the black box and three extra ports on the wireless router.

use the following commands in the QNX terminal to log into the vehicle:

#ssh -l vehicle gcrear0

when prompted for a password use: "vehicle"

To confirm connection, the "ls /net" command should display "baja" and "gcrear0" (NOTE: if "gcrear0.overbot.org" is displayed then refer to the vehicle network settings).

2.1. *Create Waypoint File*

In manual mode position the vehicle at the beginning of the desired GPS waypoint course.
Start the watchdog on the computer using the command: "watchdog startfilebasic.txt > /tmp/logs/logfilename.txt"
The name of the logfile and the directory can be changed to the directory of your choice.
Open the GPSINS gui on the right side toolbar.
In the gui select file>open way point file
Save the Waypoint file name in a home directory /home/nagle/waypointfile.txt
At each desired waypoint, select File>Add Waypoint.
After the last waypoint is entered, select File>Close Waypoint File.

2.2. *Edit the Waypoint File*

Open a terminal and become the root user.

#su //enter the root password "foo00baz"
#ped /tmp/logs/logfilename.txt

The format of the waypoint file is as follows:

Latitude | Longitude | Horizontal Allowance | Speed Limit

The defaults for the horizontal allowance and the speed limit are 13.30 ft. and 15 mph respectively. Sometimes the Overbot needs more than the default 13.30 feet for the first waypoint. Usually the last waypoint should have a speed limit of 0 mph. Edit the horizontal allowances and speed limits in the waypoint file and then save the changes.

2.3. *Move the Waypoint File*

In the same terminal, copy the waypoint file over to the vehicle computer

#cp /home/nagle/waypointfile.txt /net/gcrear0/home/vehicle/waypoints/waypointfile.txt

3 HOW TO RUN THE OVERBOT AUTONOMOUSLY

Open the QNX terminal
log into the vehicle (see how to ssh above) and run usercontrol
#usercontrol

In usercontrol select the desired waypoint file

In usercontrol set the vehicle to active

Use the E-stop remote and select RESET/ON

Use the E-stop remote and select PAUSE

Switch the vehicle into AUTO using the panel to the left of the steering-wheel

In usercontrol select “reset” and then “exit”

Stand clear of the vehicle and hit RUN on the E-stop remote.

***The vehicle can be stopped at any time by pressing the PAUSE button on the E-stop remote. It can then start back up after being paused by pressing the RUN button.

When the vehicle gets stuck or finishes the course successfully, be sure to hit the PAUSE button on the E-stop before you approach it. Check to see if the transmission has been put into neutral (this is typically a sign that the Overbot has finished its mission). Switch the MAN/AUTO to MAN using the panel to the left of the steering-wheel. Release the brakes fully and check to see if the Overbot light has stopped flashing. In user control select “standby” to stop the watchdog.

4 MODIFICATIONS MADE TO THE OVERBOT

4.1. *BLOWER FANS*

The 24V marine blower fans are installed on the top of the truck-bed. They are located directly above the generator. The generator sucks air and in result dust tends to be sucked into the truckbed from below the vehicle. The purpose of the blower fans is to equalize the pressure to reduce the amount of debris entering the truckbed. The blower fans are spliced into the 24V power bus in the main power box. The power wires for the fans are routed towards the right-hand side of the vehicle by the back of the black computer box.

4.2. *AUTONOSLUG BOX*

General Concept:
The autonoslug box uses two 555 timers, some NAND gates, and an NPN transistor to crank the generator for approximately 3 seconds, then wait 15 seconds and repeat this cycle until the generator is started. At any time if the STOP signal input is triggered on the device, the 555 timers' reset pins are triggered and the device will not crank until the entire system is powered down and powered back up.

It has three inputs:

1. POWER – 12V from the generator battery (only applied when aux battery switch is in the on position)
2. RUN signal – 12V signal from the generator
3. STOP signal – 12V signal from the kill-switch

And one output:

CRANK – signal tied to the remote start crank of the generator. This cranks the engine when it is tied to ground.

The voltage of all three inputs is stepped down to 5V using 12V to 5V regulators. Both 555 timers are setup in a monostable configuration using one resistor and one capacitor to achieve a specific waiting time. After pin 2 is triggered, the timer waits *T* seconds before the output pin 3 is triggered, where:

$T = 1.1 \times R \times C$ (*R* is the resistance in ohms and *C* is the capacitance in farads)

The first 555 timer cranks the generator for approximately 2.97 seconds after which the second timer stops the crank and waits 16.5 seconds. The logic will continue to loop

through this cycle of cranking and waiting until either the generator starts or the kill-switch is hit.

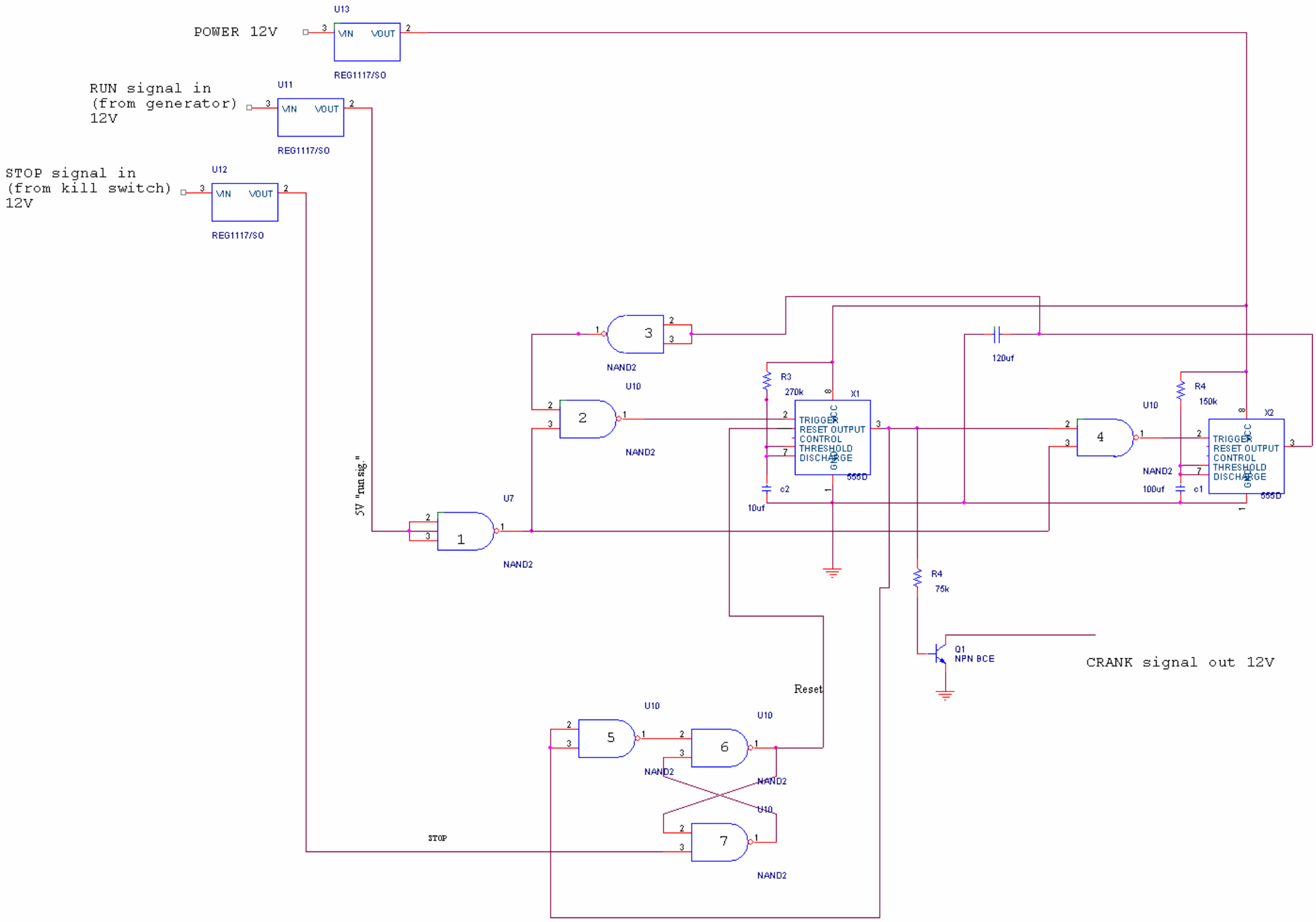
Once the generator has started, the unit will not crank. If for some reason the generator stops (not as a result of hitting the kill switch), the unit should resume the crank and wait cycle until the generator starts up again.

If at anytime the kill-switch does not seem to stop the unit from cranking the generator, the switch to the aux battery can be turned off to stop the device as well.

For more detailed information, refer to the schematic of the unit on the following page

Note: All Regulators are 12V to 5V

AUTONOSLUG AUTO-START UNIT



4.3. *AC 120V OUTLET*

The AC outlet on the rear of the main power box in the back bed is now powered. It is wired such that the outlet is powered when the vehicle is running off of the generator or the AC plug power.

5 CVS Server

The CVS Server is currently up and running on the overtux server. A few good references on how to use CVS are:
http://owen.sj.ca.us/~rk/howto/cvs.html#access
http://www.network-theory.co.uk/docs/cvsmanual/cvs_14.html
http://cvsbook.red-bean.com/cvsbook.html .
Below is a reference to the key points in the CVS setup that are required for it to operate correctly.

5.1. *Password and Writers File*

To give new users access to the CVS repository, they need to have a login name and password. To add the new user you must edit the two files in the CVSROOT folder on the overtux server: /cvs/newcvs/CVSROOT/writers and /cvs/newcvs/CVSROOT/passwd
The writers file tells CVS which usernames are allowed to commit changes to the files in the repository. The password file tells CVS what passwords correspond to which user names.

If the new user needs privileges to commit changes, open the writers file in an editor and add a new line with the new user name on it. Save the changes.

The new user must be added in the passwd file. The format of the passwd file is:
Username:encryptedpassword:cvs
To create the encrypted password, use the command on the overtux server:

#htpasswd -n password username

The output will show the username:encrypted password on the standard output screen. Copy the output into the CVS passwd file so that it matches the format:
username:encryptedpassword:cvs
Save the passwd file. The new user now has access to the CVS repository.

5.2. *File Permissions*

In order to setup remote access to the CVS directory, the permissions of all the files and modules in the repository must have read and write access for users and groups. Also the

sticky bit must be set for all the modules main folders in the CVS directory. They are currently set so that after a user commits changes, the permissions stay as they should. The repository is assigned to the "cvsusers" group. To assign the correct group and set the sticky bit on the repository directories, the following commands were used:

#find cvs/ -exec chgrp cvsusers {} \;
#find $CVSROOT –type d –exec chmod g+s {} \;

These commands need not be repeated now that the permissions are correctly set. However, if the permissions get messed up somehow, you may want to check to see if the sticky bit is still on. To check this you can look at the file permissions of a folder by typing:

#ls –l /directory

This should give you the permissions of the files and directory in the order user, group, and other users.

5.3. *Lock Folder*

The lock folder is necessary for CVS to keep track of the checked out modules and keep its internal data structures intact. The lock folder directory is located in /cvs/.lock. The permissions are set so that any user can access and change the files in this directory.

5.4. *Normal CVS USAGE*

The references listed at the beginning of Section 5 are very helpful with learning CVS commands. Always make sure the environment variable on linux and qnx remote machines is set properly in the format shown below. Use this command replacing the proper loginname:

export CVSROOT=:pserver:loginname@overtux.cse.ucsc.edu:/cvs/newcvs

To login to the overtux CVS server from a remote machine, use the following command:

#cvs login

#cvs checkout <module> //use this to checkout a module from cvs

#cvs update <module> //use this to update a module

#cvs edit <module/file> //use this command to begin editing a module/file

#cvs unedit <module/file> //use this when you are finished editing

#cvs commit <module> //commits local changes to the CVS server

#cvs add <module/file> //add new files

#cvs logout //logout

6 GPS Heading Filter

We investigated different filter possibilities to apply to the GPS heading data. We were looking to smooth out the heading data and eliminate high frequency changes due to GPS position jumps. We tried using a second-order IIR low-pass Bessel filter and an FIR Hamming window. Both successfully eliminated the high frequency jumps, however there was a significant lag time and the low frequency data wasn't very smooth. We then tried a least squares filter. We think that this is the most appropriate filter for the data. This filter determines heading for any given position by applying a linear least square fit to "num" previous points up to that position. The slope of the fit is used to determine heading. I higher number of points creates a smoother heading but too many creates a observable time lag when the vehicle turns. We found num=10 to be a pretty good number of points.

The matlab code is as follows:

```
%filter heading
function filtheading(gpslog)
close all
hold on

num = 10;          %number of data points to use for the least squares fit
X = ones(num,2);       %matrix for X positions and a column of ones
Y = zeros(num,1);      %column for Y positions
T = [];           %initializing variables
d = [0 0];

for j = 1:num-1        %this loop gets the first minimum number of data points
   X(j+1,1) = gpslog(j,8); %to apply the least squares fit to
   Y(j+1,1) = gpslog(j,7);
end

for n=num:length(gpslog)    %This runs through the rest of the data
   X(1:num-1,1) = X(2:num,1);  %For each new position the X and Y data is
   Y(1:num-1,1) = Y(2:num,1);  %Moved and the new position is added to the matrix
   X(num,1) = gpslog(n,8);
   Y(num,1) = gpslog(n,7);
```

```
    P=X\Y;              %Solution to the least square fit is stored in matrix P
    slope = P(1);               %P(1) is the slope and P(2) is the intercept
    Thead = atan(1/slope)*180/pi;     %Slope of the line is converted to heading angle using atan
                                %note that the slope has to possible directions! This is taken
                               %care of below


    Thead = unwrap(Thead*pi/180)*180/pi;


    d1 = Y(num)-Y(1);       % the next three if statements determine if the heading is
                        %north/south and east/west then apply an appropriate 180
                        %correction to the heading

    if abs(Thead-90)<2
      d1 = X(num)-X(1);
    end

    if abs(Thead+90)<2
      d1 = X(1)-X(num);
    end

    if d1<0
      Thead = Thead+180;
    end

    T = [T;Thead];      %The new absolute heading is stored into a vector T
end

T = mod(T,360);         %This takes care of rewrapping the heading (not really needed
                    %but just in case heading goes over 360)

%plot(T,'g');           %plots the new heading in green



yaw = [];
for j=num:length(gpslog)
    yaw = [yaw;gpslog(j,21)];   %this gets the YAW data
end

correction = T - yaw;       %The correction is calculated using the GPS least squares heading


correction = mod(correction,360);
plot(correction,'g');
heading = correction;       %the correction is saved as the heading variable because a
                        %low-pass filter for the heading was already written, so we %recycled the code
to low-pass filter the correction instead.

V = [0 0 0];
kk = 0;
```

```
for kk = 3:length(gpslog)       %This run through the GPSlog and finds the first 3 times
            %the velocity of the vehicle exceeds 0.1 m/s
   V(1) = sqrt(gpslog(kk,13).^2 + gpslog(kk,14).^2);
   V(2) = sqrt(gpslog(kk-1,13).^2 + gpslog(kk-1,14).^2);
   V(3) = sqrt(gpslog(kk-2,13).^2 + gpslog(kk-2,14).^2);

   if V(1)>.1 && V(2)>.1 && V(3)>.1

 kk        %This reports the position at which the vehicle is considered to
          %be officially moving.
      break
   end
end

% b = [0.00094469184384   0.00188938368768   0.00094469184384];
%
% a=[1.00000000000000  -1.91119706742607   0.91497583480143];

[b,a]=butter(2,0.02,'low')      %Here the Butterworth low-pass filter difference equation
                        %coefficients are calculated.  The second variable "0.02" can be
                        %changed to alter the Wn of the filter.  (perhaps a lower Wn
                        %could be used when the GPS has fewer satellites)

fltheading = ones(kk-3,1);    %The next few if statements take care of setting up the heading
%correction vector so that it can be used in the difference %equation.

for i = 1:kk-3
   fltheading(i) = NaN;
end

fltheading(kk-1) = heading(kk-1);
fltheading(kk-2) = heading(kk-2);

for i=kk:length(heading)        %Difference equation is applied to filter the heading  correction

fltheading = [fltheading; b(1)*heading(i)+b(2)*heading(i-1)+b(3)*heading(i-2)-a(2)*fltheading(i-1)-
a(3)*fltheading(i-2)];
end

fltheading = fltheading/a(1);
fltheading = mod(fltheading,360);

plot(fltheading,'r');
```

******This can be run directly in MATLAB and is labeled filtheading.m*******
After a GPSlog file has been imported into MATLAB as a variable the m-file can be run using "**filtheading(***variable_name***)**"