Introduction:

This document will be a step-by-step guide to setting up the search demos created by Yu Chen Lim and Adam Willats. These demos are designed to be run on the willow-garage Turtlebot using two laptops. These demos rely heavily on an environment call Robot Operating System (ROS, @ www.ros.org) and currently ROS only runs on Ubuntu systems. More specifically for the turtlebot you must **install Ubuntu 11.10 as the operating system** and then use ROS-electric (fuerte is the newest ROS version but is not yet, as of 8/2/2012 running on the turtlebot). The standard setup is to have one laptop (in our case a macbook pro 13”) sitting on a lower shelf of the turtlebot from now on refered to as “the robot” or “R:”, and another laptop (in our case an asus eee pc) which functions as the workstation sending commands to the robot and receiving data. This laptop will be refered to as “the workstation” or “W:” for the rest of the report.

Notes:
-R:, the laptop on the robot will be running almost all the computation, we advise choosing a laptop with a reasonable amount of processing power. Our mackbook pro 13” did well, but was running at full CPU on a regular basis.
-We recommend gedit for most text-editing

Step 1: Install Ubuntu 11.10 on both the robot laptop (R:) and the workstation (W:)

- see **PC:** releases.ubuntu.com/11.10 **Mac:** cdimage.ubuntu.com/releases/11.10/release/
- Select an appropriate download for a **Desktop CD** , be sure to choose a 64-bit version to work well with torch7

Notes:
For Mac
-You can dual boot (run both Mac OSX and Ubuntu) using rEFIt @refit.sourceforge.net
-Ubuntu 11.10 seems to need both a bootable USB and a Live CD, if you still have problems booting from the USB try http://www.ocztechnologyforum.com/forum/showthread.php?95640-BOOTING-UBUNTU-with-USB-on-MAC

For PC:
-You can avoid repartitioning your hard drive by using Wubi, which makes it very easy to install Ubuntu inside windows! Use the wubi at the bottom of releases.ubuntu.com/11.10

Step 2: Setup a network between R: and W:

introduction: There are many ways to communicate between the two computers, but we found that this was the most stable if no router is available, and has the added advantage of a static IP

W:

- Click on the wireless icon on the top panel on Ubuntu.
- Click Edit Connections

- Click on Wireless tab
- Click Add
  - Connection name: (Any name)
  - SSID: (Any name)
  - Mode: Ad-hoc
    Leave others the same
  - Go to wireless security tab
    - Choose WEP 40/128-bit Key
    - Type your password
    - IPv4 Settings: Shared to other computers
  - To start the connection, click the wireless icon on the top panel
  - Click Connect to Hidden Wireless Network
  - Select the network you have created, and click Connect

R:

- Connect to the ad-hoc network like any normal Wi-Fi network

Notes:

- The robot computer may not see the network when the workstation computer has created it, because the Wi-Fi network list is still not refreshed. In that case, just click on the wireless icon, uncheck enable wireless, then check it back, and you should see the newly created network
- Type **ifconfig** in a terminal window to see the IP info of the computer
  - **look under the 'wlan0' section, and note down the number next to 'inet addr' this is your IP for that computer**
- The wireless card on Mac does not work on a fresh Ubuntu install. To install the driver package, go to
  http://askubuntu.com/questions/109983/macbook-pro-8-1-wireless-interface-not-showing-up (If the kernel being used is pae version, instead of generic on the backport package, use generic-pae)

Step 3: Setup Remote Connection

introduction: this will allow you to access R: from W: to run commands while R: is closed on the robot.

- install ssh on R: and W:
  - do this by running **sudo apt-get install openssh-client** and **sudo apt-get install openssh-server** on both computers!
- test out ssh!
  - Try running, from W: **ssh username@R_IP -X** (where 'username' is the name of the account on R: and where 'R_IP' is the IP from R:. (note to find R_IP follow instructions in the notes of Step 2!)
  - then try messing around with their files!
  - If ssh fails, diagnose the connection, execute **ping R_IP** to see if a connection exists
- On R:, open Desktop Sharing, check “Allow others to view your Desktop”, uncheck “You must confirm each access to this machine”, and put a password for security purpose.

NOTES:

- the '-X' flag is required to run graphical applications (such as gedit) from the second computer

Step 4: Setup ROS on R: and W:

Simply follow the 3 installation tutorials @ www.ros.org/wiki/Robots/TurtleBot
(see Notes)

Verify everything's working, by reading through and trying out “TurtleBot Bringup”, “TurtleBot Teleoperation:”, “TurtleBot Odometry and Gyro Calibration,” “TurtleBot Follower Demo”

Notes:

- Whenever the tutorial refers to “install on the workstation” that is W: in your setup (in our case the asus eee pc),whenever the tutorial refers to “install on the TurtleBot” that is R: in your setup (in our case the macbook pro). Note, nothing gets installed on the physical hardware of robot, all commands are sent to the motors from “the robot” which is actually the laptop sitting on the base.

- for tutorial 1 - Install Software on the Turtlebot
  - if you are not using the prepackaged laptop for R: (we suggest you do use your own laptop), then follow www.ros.org/wiki/Robots/TurtleBot/Robot%20Install
- for tutorial 3 – TurtleBot Networking Setup
  - if you are not using the prepackaged laptop for R: (we suggest you do use your own laptop), then follow www.ros.org/wiki/Robots/TurtleBot/Robot%20Install
- for TurtleBot Odometry and Gyro Calibration
  - If you start the robot using minimal.launch, the gyro calibrations you make will not take effect once the computer has shutdown, unless you paste
    <param name="turtlebot_node/gyro_scale_correction" value="???"/>
    <param name="turtlebot_node/odom_angular_scale_correction" value="???"/>
    into the file minimal.launch after running the command 'roscd turtlebot_bringup' to take you to the correct directory. Of course, you should replace “???” with your calibrated values!
- If you have a recurrent warning about not being able to read laptop battery, go to this link http://answers.ros.org/question/30631/turtlebot-unable-to-check-laptop-batterybat0state/ and follow the instructions. (laptop_battery.pyc might need to be removed as well) When you launch minimal.launch, if there is still an error about minimal.launch, then change the permission of laptop_battery.py to the same as those on the same directory.

Setup 5: Install torch7 & online-learner

- Go to https://github.com/andresy/torch and follow the instructions there to install torch
- Go to https://github.com/e-lab/online-learner and follow the instructions there to install the prerequisites for the online-learner (OL)
  Note: The modules should be installed using sudo torch-pkg install (module name)

instead of luarocks. The module camera, nnx and parallel should be installed as well). The parallel module requires the Ubuntu package libzmq-dev

- install online-learner (OL) by executing 'git clone https://github.com/e-lab/online-learner.git' in a terminal
- then clone our specific branch (the 'turtle' branch) using an approach from here http://stackoverflow.com/questions/1911109/git-clone-a-specific-branch
- To install the libfreenect wrapper for Torch 7, follow the instructions at https://github.com/benoitcorda/torch7-kinect.
  Note: For the link in the instructions, do an Ubuntu manual install.
- Then, clone the torch7-kinect repo down by doing
  - **git clone https://github.com/benoitcorda/torch7-kinect.git**
  - and do
  - **cd torch7-kinect; mkdir build; cd build; cmake ..; make; sudo make install**

  Note: cmake should throw an error indicating that libfreenect is not installed if you are using a 64-bit system. In that case, go into cmake_modules/FindFreenect.cmake, and add /usr/local/lib64 under FIND_LIBRARY
- test the online-learner by running:
  - **cd ~/online-learner;**

Notes:

-PLEASE clone the online-learner (i.e. run the git clone command after running 'cd ~')into your home directory, the path to the online-learner is hard-coded into many of our algorithms. If you clone it somewhere else, the programs will not run until you go through and manual change EVERY reference to the online-learner;

- Some suggested starting navigation parameters to change:
  In turtlebot_apps/turtlebot_navigation/config open up base_local_planner_params.yaml,
    change max_rotational_vel to 1.0,
    min_in_place_rotational_vel to 0.4,
    acc_lim_th to 3.2
  In amcl_turtlebot.launch,
    change the odom_alpha1 from 0.2 to 0.5

Setup 6: Install our search demo files

- change directories to home, by executing 'cd ~' in the terminal
- clone the turtlebot repo into your local folder using
  'git clone https://github.com/e-lab/turtlebot.git'
- now check whether you already have a ros_workspace folder inside home. If so, simply cut the contents of the newly made ~/turtlebot folder and paste them into ros_workspace.
  - Check for 'ros_workspace' by executing
    cd ~; [-d ~/ros_workspace]&&echo "exists"||echo "not exists"
  - if you see “not exists” jump to the “Otherwise” section a couple of lines down
  - if you see “exists,” then you'll simply need to copy the contents of ~/turtlebot into ros_workspace. You can simply drag the files in using nautilus (the file explorer)

or by executing the following code
    - **cp -r ~/turtlebot/* ~/ros_workspace**
  - you then need to delete the turtlebot folder, again either from nautilus, or by executing
    - **rm -r ~/turtlebot/*; rmdir ~/turtlebot;**

- *Otherwise,* if you don't have a preexisting ros_workspace folder, simply rename the ~/turtlebot folder to ~/ros_workspace
  - this can be done using the command-line by executing
    mv ~/turtlebot ~/ros_workspace

Everything should now be setup! Congrats on making it this far! Now time to test stuff out!

Test 1: Saving a prototype image

Note: we recommend that all images be taken with the same camera you'll be using later in the full demo. In our case this was a USB webcam mounted on top of the turtlebot

R: (using remote desktop from W:)
- start up the robot
  roslaunch turtlebot_bringup minimal.launch

- change into the online-learner directory
  cd ~/online-learner

- run the online-learner
  sudo torch run.lua -d 4 -b 184 –source=kinect
  - **Note:** if the above doesn't work the first time, close it and try again. For some reason, the online-learner sometimes needs one warmup run.

- Clear previous prototypes
  - click “forget all”

- place an object of interest in view!
  - We suggest an object with colors and features unlikely to be found elsewhere in the background image
  - we also suggest an object which is large enough to be easily resolved (more info on that later)

- take a prototype image!
  - Hover your cursor over the object somewhere in the image
  - If your object does not fill the blue circle by your cursor (or at least mostly fill it), please move the object closer or choose a larger object. Also make sure the blue circle does not extend past the edge of the image!
  - With the object inside the circle, click once on the object to add a prototype image. Here you can take a couple of images from different angles if you want. More images means the OL is more flexible at recognizing the object, however too many images will make the program slow and possibly cause false-positives

- save the protoypes!
  - Click “save session” button at the bottom of the screen

NOTES:

-d 4 -b 184 are flags which downsample the image, allowing the program to run at a higher framer right (with a loss of image resolution). These downsampling flags are also hardcoded into the later demos, which is why we recommend using them now, when the prototype is being taken. If you later change the main demo to downsample more or less, adjust the flags on your prototype capturing accordingly

--source=kinect tells the program to use the kinect RGB camera for the video feed, however you can run the program without this flag to use the built in webcam, or with – c 1 or -c 2 to choose another usb webcam to use

the object has to fill the blue circle to decrease the amount of background image in the prototype. Essentially you only want the program to remember the object, not what happened to be on the wall behind it

Test 2: Obtaining a map

Before using the node-based search algorithm, a map of the space to explore needs to be obtained. There is a tutorial about building a map using SLAM for TurtleBot. http://www.ros.org/wiki/turtlebot_navigation/Tutorials/Build%20a%20map%20with%20SLAM. Make sure the map is not saved in the /tmp folder, as it will be erased.

Test 3: Positioning in a map

After obtaining and saving the map, the map should be loaded and the robot should know about its initial position in the map. The amcl package should be used and the link to to the tutorial is http://www.ros.org/wiki/turtlebot_navigation/Tutorials/Autonomously%20navigate%20in%20a%20known%20map. To use the node-based search algorithm, just follow the tutorial until 3.3.1 Localize the TurtleBot.

Test 4: Simple Search

To launch the simple search algorithm, prototypes of the object to be recognized should be saved. If you do not have prototypes for the object, see the section “Test 1: Saving a Prototype Image”. If you have prototypes, then just follow the steps below:

- Connect to R: remotely through VNC
- Open up a terminal in R:, enter “roslaunch turtlebot_bringup minimal.launch” to start up

the TurtleBot

- Then, enter “roslaunch turtlebot_bringup kinect.launch” to start up the Kinect
- Then, you are ready to launch the search algorithm. Enter “roslaunch simple_search simple_search.launch”
- Then, the robot should start to search for the object.

Notes:

- If you see a “blanked out” message on the terminal after launching the search algorithm, it is caused by the laser scan data being invalid, which may be caused by Kinect being too close to an obstacle, or just bad data. Currently, the algorithm currently checks for a few consecutive invalid data, and treats it as an obstacle. Restarting the Kinect (i.e. power off the robot and unplug the Kinect's USB from the computer) might decrease the occurrence of this problem.

Test 5: Node-Based Search

– Make sure to run Test 1, 2 and 3 to save a prototype and place yourself in a map.

– You do not have to redo Test 1 everytime, if you want to use the same image.

– Then in R: execute **roslaunch robot_search robot_search.launch**

Important Final Notes:

if anything goes wrong with the programs or you want to shut everything down, you can shut down all ROS programs by executing **killall python** on R:. After killing all ROS processes, you will need to run **roslaunch turtlebot_bringup minimal.launch** to start up ROS

We aliased many of the commands such as “roslaunch turtlebot_bringup minimal.launch”. Check out a tutorial on linux aliases to save yourself some time and typing!

There may be many bugs and problems that you encounter. Please direct ROS questions to ros.org, however questions you feel are specific to our algorithms/tutorials please direct to https://github.com/e-lab/turtlebot

Good Luck!
Adam Willats & Yu Chen Lim
Purdue e-lab
Summer 2012