4 Haptic Interface – Tanvas

The users commanded the swarms using a TanvasTouch monitors from the company Tanvas. The TanvasTouch monitors are capable of rendering textures, which we use to help the user better orient themselves in their environment. To send a command to the ergodic controller, users double-tap, shade the region of interest, then double-tap again.

4.1 System requirements

To use the TanvasTouch, you must use a Windows OS and Visual Studio (VS). For touch detection, we use the Universal Windows Platform API. It may be necessary to download additional VS packages to use Universal Windows Platform. For reference, we used Windows 10.0 build 18362 and Microsoft Visual Studio Community 2019. The ROS side of the TCP socket (in package user_input for the human subject tests or tanvas_comms for field tests) was created on ROS melodic.

For networking, both the TCP server and client must be connected to the same router and be connected to the same IP address and port defined as a variable in the top of both files. It is best to use a private network in which you have access to the network settings. If you experience problems, you likely need to connect to a static IP address and/or disable the security features on the computer.

When you first receive your TanvasTouch, follow all of the instructions for setting up the device provided by the company, including setting the "Main display".

4.2 Tanvas code for FX3

This code was used during FX3 so that a human input could be used to drive the ergodic control algorithm. It is a pared down version of the code used in the human subject experiment at mschlafly/darpahaptics.git.

Information gets transferred between programs using the txt file "touch_locations.txt". You can "hack" the system for testing or training by changing the values in the txt files.

Install our code:

1. Open the Git Bash application from the Start menu.
2. Navigate to the folder you would like to keep this repo. For example, you could use "cd ./Desktop" to place folder on your desktop. "cd .." is used to move up a folder.
3. Clone repository from git: "git clone https://github.com/mschlafly/darpademo.git"

4.2.1 Descriptions of scripts

- TouchDetection - This program utilizes the Universal Windows Platform touch detection C# SDK for app development to record the locations of user inputs. Input recording starts and ends with a double tap. The double tap to end recording also saves the data to text file "touch_locations.txt" in the Downloads folder on the computer. This program would contain any image to be displayed in the MainPage.xaml file. Note, if you would like to use your own image, you must include it in the project by going to the Solution Explorer, clicking "Show all files", then right clicking the file and chosing "Include in Project"
- TCP_socket - Server side of a TCP socket using the winsock library. Multiple robots can connect and receive messages from this server. This code (a) creates a TCP socket for communicating between a windows OS and linux OS, (b) connects to client on linux OS and tests the socket, (c) continuously sends over coordinates for the touch input (read from "touch_locations.txt" in the Downloads folder). The coordinates are send over in batches to avoid any potential loss of data. Our project-specific code can be found within the function EchoIncomingPackets in threaded-server.cpp.
- tanvas_comms (In the ROS repo) - ROS package that is run on the linux OS for communicating over the TCP socket and publishing touch input locations. This code can be found in the ROS codebase for the human swarm collaboration simulator.

4.2.2 Instructions for running code

1. Connect the TanvasTouch (all 3 cords), open the tanvas engine application, and calibrate monitor by pressing "Calibrate".

2. Connect to static IP

   (a) connect to private network

   (b) turn off any and all firewalls (on XPS laptop using Mcafee application)

   (c) set static IP by going to the control panel through the Start menu, clicking "Network settings and tasks" then "Change adapter settings", left click on "Wi-Fi" and select "Properties", "Internet Protocol version 4 TCP/IPv4", then "Properties" again.

   (d) Select "Use the following IP address" and set the IP address to the one used at the top of the TCP socket code (192.168.1.3 in our case) and the subnet mask should auto full to 255.255.255.0.

3. Open the touch detection code in VS from the solution file, either "ergodicinput.sln" or "waypointinput.sln", and the TCPserver code from solution file "TCPserver.sln". Access the source code through the Solution Explorer.

4. The TCPserver will need to be run from the command line. However, accessing the C compiler though VS requires a couple of steps. (Instructions are also provided at the top of main.cpp.)

   (a) To access VS terminal, go to Tools > Command Line > Developer command prompt

   (b) To compile: "cl -GX threaded-server.cpp main.cpp ws-util.cpp wsock32.lib" (This is usually not necessary)

   (c) Go into file TCPserver "cd ./TCPserver" and check that that contains the cpp files using "dir".

   (d) To execute: "threaded-server.exe 192.168.1.3 8888" where 192.168.1.3 is the IP address and 8888 is the port. These IP addresses and ports are specific to our set-up.

5. Make sure the same IP address and port are defined at the top of client.cpp. After sourcing, using catkin_make and roscore, the node can be run with "rosrun tanvas_comms client" or from a launch file.

6. Copy and paste the files (sound1.wav, sound2.wav, and "touch_locations.txt") in TouchDetection_demo to the Downloads folder. (The Downloads folder is used so that you do not have to edit the file permission settings on your computer. It may be helpful if you emptied your Downloads folder as well.)

7. To run the touch detection app "sharedcontrol", find the play button at the top and click it. Make sure that the drop down menus next to it reads "Debug" and either "x64" or "x86". When this file is run, you will be asked to pick files from the Downloads folder.

8. Make sure the app is shown on the main screen. Send commands by double tapping, shading the region of interest, and double tapping again to send the message. The locations of the taps are included in the message. You should hear a "beep" after every recieved double-tap (if your computer sound is on).