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 human subject tests

This code was used during during the human subject tests to receive human inputs from the user, display haptics so that the user can orient themselves in their environment, and decide where to send commands, and sends the command over a TCP socket. The client side of the socket can be found at https://github.com/mschlafly/VR_exp_ROS.git.

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

A previous version of this code switched between global and local modes and displayed buildings and parks haptically. These features were removed to make the system less complex to use. Look on github to find those previous commits.

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/darpahaptics.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. To send a waypoint command, you also need to tap once to send drone one to that path/points, twice to send drone 2, and three times to send drone 3. The locations of the taps must be in the same location of the last double-tap. Audio will be played after every detected double-tap and tap.
  - The TouchDetection folder contains VS projects for an ergodicinput (a.k.a. shading an area of interest) and a waypointinput (a.k.a. giving waypoints and/or paths for the drones to follow). The remaining files are selected from the filepicker (after being moved to the downloads folder).

  - 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 chose "Include in Project".

- HapticDisplay - This C# program generates haptic sprites representing objects in the unity environment on the TanvasTouch monitor display. The position of the player and rotation of the aerial map change to correspond to the minimap shown in the Unity environment. This information is read from the text file "person_position.txt". It is based on the HelloTanvas demo provided by Tanvas. A previous version of this code updated sprites corresponding to building and bush locations according to the person's position for a zoomed-up view of the map. To access this code and see how this was done, look at previous commits.

- Create_haptic_features - Haptic objects are created based on a bitmap provided in a png file where a black pixel is max friction, a white pixel is no friction, and grey pixels are somewhere in between. This folder contains python scripts that create these images, one for the person in person.py and one for the boundaries in make_global_display.py. make_global_display.py also creates a visual for training purposes (although the visual is a little off due to an incorrect transformation in the code). Previous versions of this folder had code for generating haptic sprites for buildings, bushes, and corresponding training examples. Please view past code and README instructions to view that. The images are saved in Haptic_display/HelloTanvas/Assets.

- 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 (if new input read from "touch_locations.txt" in the Downloads folder) and receives coordinates for the person's position and orientation in unity, writing them to a text file ("person_position.txt" in the HapticDisplay folder) if the position has changed. 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.

- user_input - ROS package that is run on the linux OS for communicating over the TCP socket. This code be found at "https://github.com/mschlafly/VR_exp_ROS.git".

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, "sharedcontrol.sln", the TCPserver code from solution file "TCPserver.sln", and the haptic display code from solution file "HelloTanvas.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 this project.

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 user_input client” or from a launch file.

6. Copy and paste the files (begin001.wav, done0001.wav, sound2.wav, and “touch_locations.txt”) in TouchDetection 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. Change the variable docPath_person at the top of MainWindow.xaml.cs within the HelloTanvas project to the location of “person_position” on your computer.

8. To run the haptic display app, find the play button at the top of the VS project and click it.

9. To run the touch detection app (either ergodicinput or waypointinput), find the play button at the top of the VS project 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.

10. Make sure the app is shown on the main screen. Send ergodic 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. Send waypoint commands by double tapping, drawing a path that you want the drone to follow, double tapping again, then tapping to indicate which drone you want to send. You should hear a “beep” after every received double-tap or tap (if your computer sound is on).

4.3 Things to try if code is not working properly

Problems with the TCP socket:

- Check that firewalls are off, all computers are connected to the correct network, and the right IP addresses and ports are set.
- Check that “touch_locations.txt” in the downloads folder does not have a ton of numbers in the file. Sometimes when the file is too large, it will take too long to read, and the TCP socket will close. The contents of this file can be deleted manually.
- Try pressing Ctr C in the TCP server terminal (if messages have stopped coming out). There is some kind of windows problem where a program will stop running and pressing Ctr C gets it to continue.
- See if the computers can ping each other.
- Login into your router to check to see which computers are connected.

Tips for tap detection: Getting the Tanvas to recognize a tap can sometimes be tricky. Here are some tips.

- Recalibrating using the tanvas engine.
- Pay attention to the frequency of your taps for both the double-tap and the taps. The taps are slower than the double-tap.
- Keep your finger close to the screen.
- Even though you must tap slower than you might want to, don’t leave your finger on the surface of the tanvas for too long.
- Hitting the tanvas harder doesn’t help.

Tips for haptic recognition:

- Move your finger lightly across the surface. Putting too much pressure on the surface will cause your finger to get sweaty, making the haptics more difficult to sense.
- To sense the direction a haptic object is oriented, move your finger back-and-forth along its edge.
- Visuals make it easier. Although we did not use visuals during the experiment, we did use an example for training.

Finally, try restarting the programs or the computer.