Development and Optimization of a Multimodal Natural User Interface for Patients with Severe Motor Disabilities

Introduction and Motivation

- A quadriplegic is a person who is paralyzed from the neck down.
- Per the 2010 census, 250,000 Americans live with quadriplegia [1]
- Because of their disability, they can often feel socially isolated [4]
- Computers games help us communicate and connect while also having fun
- Quadriplegics, because of their disability, cannot use the computer in the conventional manner. [4]
- **If a technology was developed to allow quadriplegics to play computer games, they may be relieved of some of their social isolation.**

Problem Statement and Success Criteria

Problem Statement: Quadriplegics are not able to use the mouse or keyboard and hence cannot use the computer to play games.

User Constraint: C1—C3 quadriplegics cannot use muscles below their neck. Only facial gestures and eye movements work reliably

Game Design Constraints

	POV	Single-Point Input
Examples	Portal 2, Minecraft	Chess, Angry Birds
Required Keyboard Functionality	8 commands: W, A, S, D, SPACE, LEFT CLICK, RIGHT CLICK, MIDDLE MOUSE	Press/Movement Press/No Movement Press/Movement No Press/No Movement
Required Mouse Functionality	Pan Mouse	Move mouse directly to desire coordinates

Keyboard Success Criteria

Category	POV	Single-point input
Functionality	8 commands	1 command
Accuracy (error)	<1.5x	<1.5x
Time to Complete	<1.5x	<1.5x
Cost	<$1000	<$1000
User Exp. (Mins. w/o discomfort)	10 mins.	10 mins.

Mouse Success Criteria

Category	POV	Single-point input
Functionality	Pan mouse in x/y	Set mouse position to desired coordinates
Accuracy (error)	<1.5x	<1.5x
Time to Complete	<1.5x	<1.5x
Cost	<$500	<$500
User Exp. (Mins. w/o discomfort)	10 mins.	10 mins.

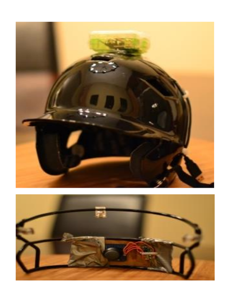
Alternate Solutions

Devices on the market

Market Survey
Eyegaze Edge
NoHands Mouse
InfoScan TS Elite
Jouse2
TrackerPro
Camera Mouse 2010

The most commonly used devices today involve a sip-and-puff controller or a modified joystick as input for an onscreen keyboard

Image Credit [5]



Wearable devices prototyped in 2013-14

A sip-and-puff controller. Credit: Adaptive Switch Laboratories

Image Credit [6]

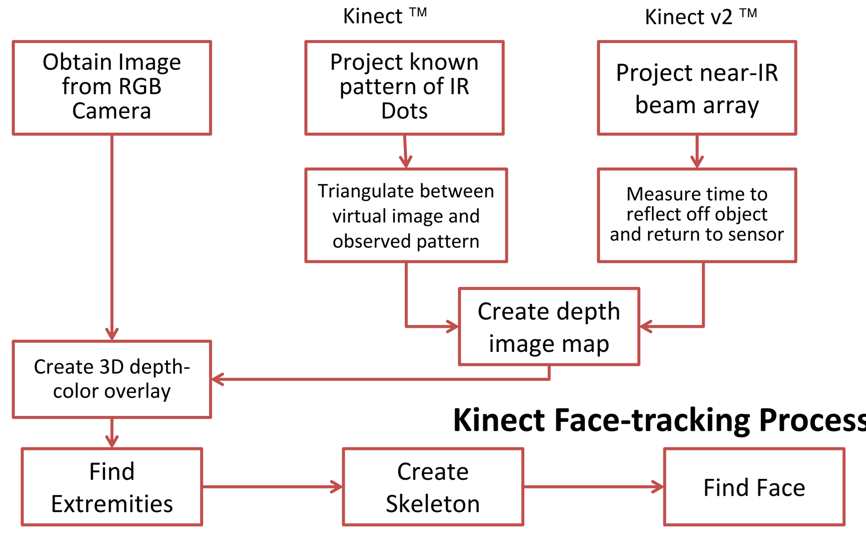
Alternative Development Options

Type	Sensors	Human interface	Physical mounting	Advantages	Disadvantages
Wearable Device	MEMS Gyroscope	Head movement	Headphones, Cap, Helmet, Eye glass	Very reliable, lowest cost	User must wear device, C1 – C4 patients cannot do large head movements, Leads to user fatigue
	Potentiometer	Tongue movement	Helmet face mask	Very reliable, clean data, lowest cost	User must wear device, Leads to user fatigue
Natural User Interface	Microphone	Voice	On PC	User does not have to wear a device	Too slow for game play. Repetitive commands are cumbersome
	Kinect™– RGB + IR camera	Facial gestures	On PC	User does not have to wear a device, usable by C1-C6 patients	Lighting must be optimized
	Eye Tracking	Eyeball movements	On PC	User does not have to wear a device, usable by C1-C6 patients	Requires focus and limits user's field of view

In 2013-14 wearable devices were prototyped and tested. In 2014-15 a Natural User Interface based on recognizing facial gestures and eye tracking will be prototyped and tested.

Background Research—Microsoft Kinect™

Kinect™ sensor array. Image Credit [7]



Kinect Face-tracking Process

Kinect™ and Kinect v2™ Attributes

Model	Color Camera res.	Depth Camera res.	Depth Sense method	Min. Depth Distance	Vertical Field of View	Cost
Kinect™	640x480	320x240	Spread Dots	40cm	43˚	$299
Kinect v2™	1920x1080	512x424	TOF	50cm – 10cm	60˚	$199

Face-Tracking Development

Task	Options	Chosen Solution
Choose facial data to collect	Raw Facial Data (1000 points), CANDIDE 3 Animation Units	CANDIDE 3 Animation Units-pre-processed data allows for high degree of accuracy
Process facial data	Hardcode expected values, use machine learning	Machine learning - Device will trainable and customizable
Send key cmds to game	C# SendKeys libraries, PIE sketch	PIE sketch - SendKeys is not compatible with games using DirectX protocol

Animation Units
Jaw Opener
Left/Right Eye Blinking
Lip Pucker
Lip Stretcher
Right/Left mouth corner pull
Right/left check puff
Right/left Eyebrow Raiser
Upper Lip Raiser
Mouth Corner Depressor

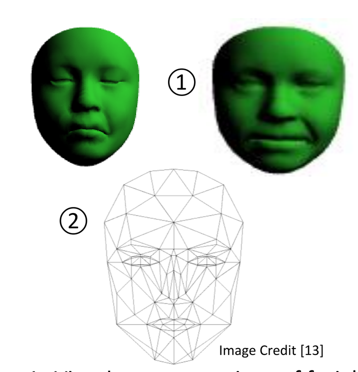
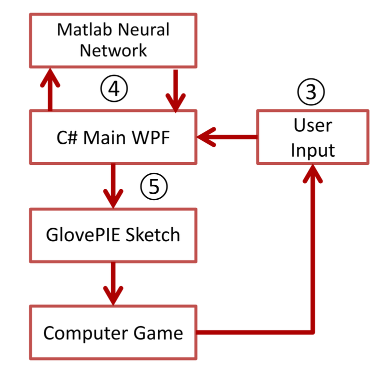


Image Credit [13]

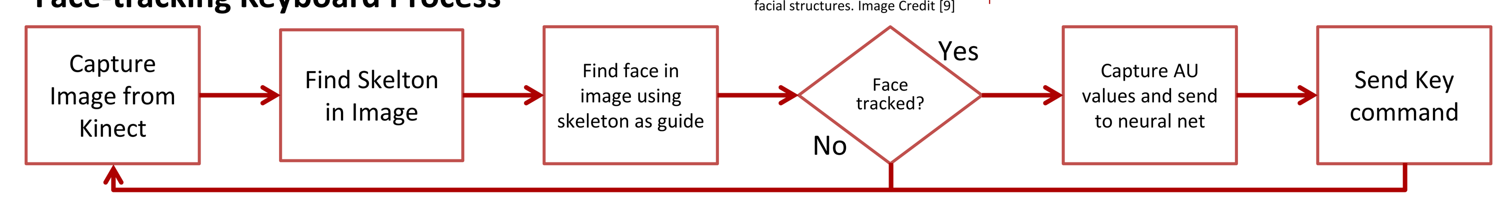
1. Visual representations of facial data collected from two test subjects
2. All face data is warped to fit CANDIDE 3 wireframe



3. User does facial gesture
4. Neural network interprets facial data
5. PIE sketch outputs command

The full interface. Image Credit [8]

Face-tracking Keyboard Process



- Two-step process to find the best animation units (AUs) for the person's facial structure
  1. Eliminate AUs which the user physically cannot execute
     - Ex. Woman with a narrow face—9 AUs eliminated
  2. Compare AU values when executing gesture to neutral face, prefer large value changes. Ex. Comparing "Jaw Open" AU value when jaw is open vs. closed

Kinect Output Values →

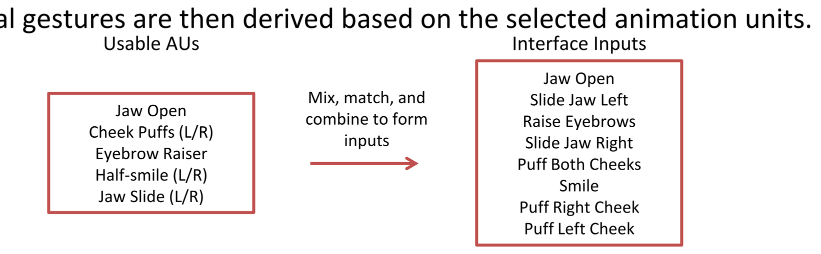
User Input Gesture ↓

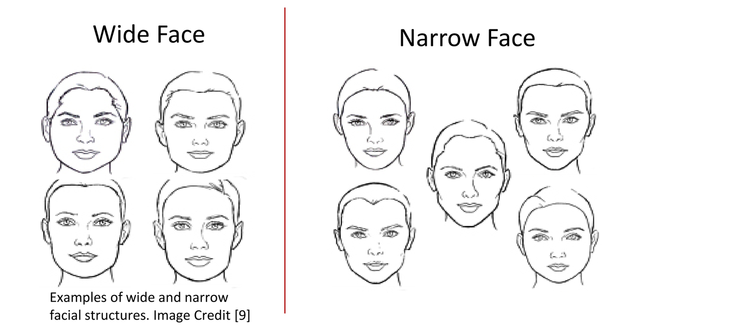
Inputed AU	Jaw Open	Jaw Slide right	Left Cheek Puff	Left Eyebrow Raiser	Left Half-smile	Right half-smile	Right cheek puff	Right Eyebrow Raiser
Neutral Face	0.01	0.02	0.13	0.22	0.01	0.23	0.01	0.12
Jaw Open	0.72	0.01	0.37	0.16	0.49	0.94	0.36	0.20
Jaw Slide Right	0.17	0.26	0.17	0.47	0.01	0.82	0.04	0.06
Cheek Puffs	0.07	0.02	0.39	0.17	0.01	0.15	0.12	0.18
Eyebrow Raiser	0.00	0.17	0.45	0.29	0.39	0.94	0.10	0.06
Smile	0.09	0.02	0.25	0.70	0.01	0.20	0.01	0.54

AU value for neutral face

AU value for executed gesture

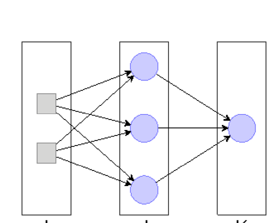
- Facial gestures are then derived based on the selected animation units.





Examples of wide and narrow facial structures. Image Credit [9]

Background Research—Neural Networks



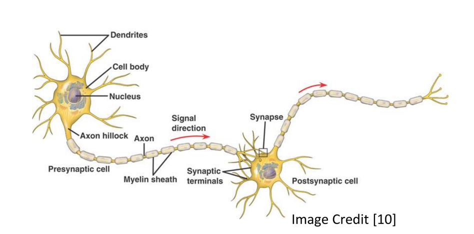
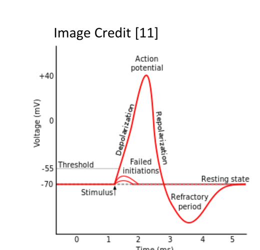


Image Credit [10]

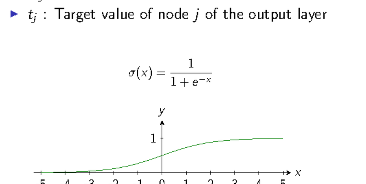
Image Credit [11]



Notation

- x_j^ℓ : Input to node j of layer ℓ
- W_{ij}^ℓ : Weight from layer $\ell - 1$ node i to layer ℓ node j
- $\sigma(x) = \frac{1}{1+e^{-x}}$: Sigmoid Transfer Function
- θ_j^ℓ : Bias of node j of layer ℓ
- $\mathcal{O}_j^\ell$: Output of node j in layer ℓ
- t_j : Target value of node j of the output layer

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Given a set of training data points t_k and output layer output $\mathcal{O}_k$ we can write the error as

$$E = \frac{1}{2}\sum_{k\in K}(\mathcal{O}_k - t_k)^2$$

- The feed-forward artificial neural network is a simplified model of the real-life biological neuron found complex nervous systems
- The accumulation of neurotransmitters in the synaptic cleft is modeled by the summation of inputs multiplied by their respective weights
- The threshold in the biological neuron action potential graph is modeled by the sigmoid transfer function

For an output layer node $k \in K$

$$\frac{\partial E}{\partial W_{jk}} = \mathcal{O}_j\delta_k$$

where

$$\delta_k = \mathcal{O}_k(1-\mathcal{O}_k)(\mathcal{O}_k - t_k)$$

For a hidden layer node $j \in J$

Image Credit [15]

$$\frac{\partial E}{\partial W_{ij}} = \mathcal{O}_i\delta_j$$

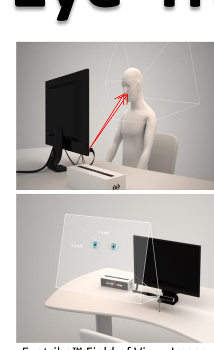
where

$$\delta_j = \mathcal{O}_j(1-\mathcal{O}_j)\sum_{k\in K}\delta_k W_{jk}$$

1. Run the network forward with your input data to get the network output
2. For each output node compute
$$\delta_k = \mathcal{O}_k(1-\mathcal{O}_k)(\mathcal{O}_k - t_k)$$
3. For each hidden node calulate
$$\delta_j = \mathcal{O}_j(1-\mathcal{O}_j)\sum_{k\in K}\delta_k W_{jk}$$
4. Update the weights and biases as follows
Given
$$\Delta W = -\eta\delta_\ell\mathcal{O}_{\ell-1}$$
$$\Delta\theta = -\eta\delta_\ell$$
apply
$$W + \Delta W \to W$$
$$\theta + \Delta\theta \to \theta$$

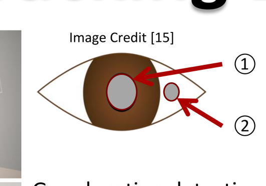
Image Credit [12]

Eye-Tracking Development



Eyetribe™ Field of View. Image Credit [14]

Image Credit [15]



Gaze location detection is accomplished by measuring the distance between the reflection of IR light off the retina (1) and the cornea (2).

Error Sources	Implications and Mitigation
Imperfect eye tracking	Noise and "jitteriness" –corrected through proper calibration
Saccades – involuntary eye movement	Sudden jumps in detected gaze – reduced through user focus

EyeTracker™ Outputs:
gazeManager.gazeData.X (x-coordinate of gaze location)
gazeManager.gazeData.Y (y-coordinate of gaze location)

Point of View games using EyeTracker™

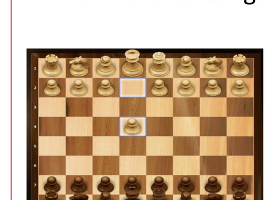
Image Credit [16b]

POV games only require the mouse to move in a general direction, as such they allow for a much greater margin of error and are less affected by error sources

Left and Up	Up	Right and Up
Left	Do Nothing	Right
Left and Down	Down	Right and Down

The screen is divided into 9 zones. If gaze is in a zone, the mouse moves in the corresponding direction

Point and Select games using EyeTracker™



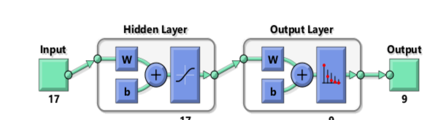
An example of a Point-and Select game. Image Credit [16a]

Point and Select games are very location dependent and have small margins of error. This makes them more affected by error sources

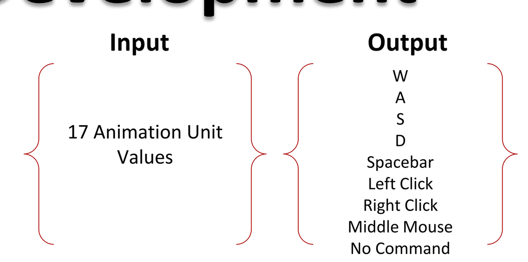
For Point and Select games, the mouse coordinates are set directly to the gaze location.

```
mouseX = gazeManager.gazeData.X;
mouseY = gazeManager.gazeData.Y;
```

Neural Network Development

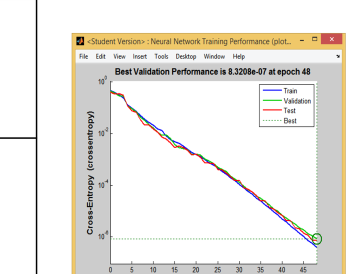


Task	Options	Chosen Solution
Program the network	Java, C#, or MATLAB	MATLAB - Well-documented , easy to create neural networks and functions
Choose type of network to use	Function approx., clustering, classification	Classification - The network must decide which of 9 expressions the user's face fits into
number of hidden layers	$0 - \infty$	1 - For most classification problems, 1 hidden layer is sufficient
number of neurons to use	Constructive or Pruning	Constructive - MATLAB auto-integrates new neurons, does not require modification of the performance error function.



Integrating Matlab Neural Network with C# WPF code

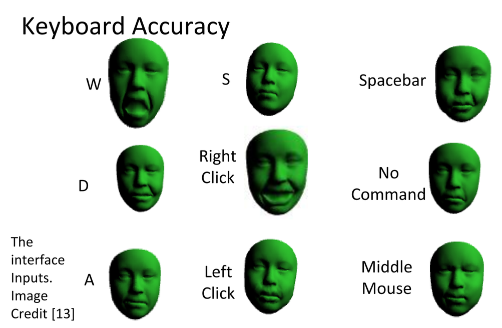
```
//initialize matlab
matlab = new MLApp.MLApp();
//Put data into the matlab workspace
matlab.PutWorkspaceData("input", "base", inputarray);
//execute the NN function nn, return result in a string.
result = (matlab.Execute("kinectv2NN(transpose(input))"));
//parse the 'result' string to get the command.
```



Neural net training with 180 samples, 70% used for training, 15% for testing, and 15% for validation

Test Setup

Keyboard Accuracy



The interface Inputs. Image Credit [13]

Each command entered 30 times, amount of correctly sensed commands is compared to amount of incorrectly sensed commands

Mouse Accuracy

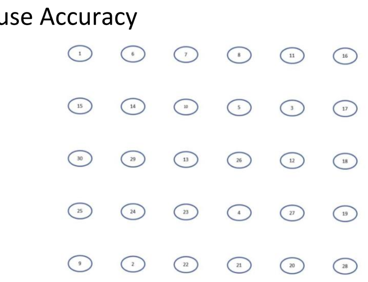


Image Credit [17]

Matrix of randomly plotted points. User must click points rapidly in order. Time and distance from target to click location is recorded

Keyboard Speed

Image Credit [16b]

Time required to play through first 5 levels of Portal 2 using face-tracking keyboard and tactile mouse

Mouse Speed

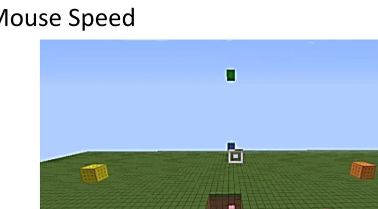


Image Credit [16c]

Test subject must use device to look at 4 targets, 30 times in a random order. The total time required to look at all 30 targets is recorded

Data

* Kinect Output Values →
User Input Gesture ↓
AU value for neutral face
AU value for executed gesture
**In number of commands correctly entered out of 30
†In seconds

Subject: Boy Wide Face

Discovery Animation Unit Data*

Inputed AU	Jaw Open	Jaw Slide Right	Left Cheek Puff	Left Eyebrow Raiser	Lip Corner Puller Left	Lip Corner Puller Right	Lip Pucker	Right Eyebrow Lowerer
Neutral Face	0.00	0.16	0.05	0.61	0.11	0.18	0.29	0.27
Jaw Slide Right	0.02	0.34	0.02	0.56	0.02	0.05	0.49	0.14
Cheek Puff	0.00	0.21	0.44	0.56	0.01	0.67	0.64	0.19
Jaw Open	0.71	0.12	0.09	0.59	0.12	0.19	0.54	0.49
Eyebrow Raiser*	0.00	0.05	0.02	0.32	0.03	0.03	0.28	0.19
Smile	0.00	0.03	0.05	0.47	0.69	0.91	0.00	0.33
Lip Pucker	0.15	0.28	0.04	0.62	0.00	0.16	0.78	0.20

Keyboard Accuracy**

Command ID	Kinect™	Command ID	Kinect™
1	30	5	30
2	30	6	30
3*	-	7	30
4	30	8	30

Subject: Woman Narrow Face

Inputed AU	Jaw Open	Jaw Slide Right	Left Cheek Puff	Left Eyebrow Lowerer	Left Half-Smile	Right Half-Smile	Right Cheek Puff	Right Eyebrow Lowerer
Neutral Face	0.00	0.00	0.01	0.26	0.01	0.01	0.00	0.37
Jaw Open	0.48	0.15	0.18	0.14	0.15	0.41	0.20	0.21
Cheek Puffs	0.01	0.02	0.10	0.65	0.10	0.21	0.03	0.55
Eyebrow Raiser	0.01	0.06	0.03	0.44	0.02	0.06	0.02	0.38
Smile	0.00	0.04	0.02	0.34	0.70	0.92	0.01	0.39
Jaw Slide Right*	0.06	0.43	0.01	0.13	0.04	0.25	0.02	0.33

Command ID	Kinect™	Command ID	Kinect™
1	30	5	30
2*	-	6	27
3	30	7	28
4	30	8	28

Subject: Boy Narrow Face

Inputed AU	Jaw Open	Jaw Slide Right	Left Cheek Puff	Left Eyebrow Raiser	Left Half-smile	Right half-smile	Right cheek puff	Right Eyebrow Raiser
Neutral Face	0.08	0.03	0.18	0.14	0.02	0.07	0.02	0.36
Jaw Open	0.75	0.05	0.11	0.11	0.02	0.30	0.02	0.40
Jaw Slide Right	0.22	0.11	0.19	0.15	0.02	0.11	0.04	0.29
Cheek Puff	0.00	0.03	0.63	0.08	0.01	0.07	0.40	0.20
Eyebrow Lowerer	0.07	0.06	0.03	0.23	0.05	0.03	0.02	0.05
Smile	0.16	0.11	0.36	0.25	0.97	0.99	0.05	0.41

Command ID	Kinect™	Command ID	Kinect™
1	30	5	30
2	30	6	30
3	30	7	30
4	30	8	30

Subject: Man Wide Face

Inputed AU	Jaw Open	Jaw Slide Right	Left Cheek Puff	Left Eyebrow Raiser	Left Half-smile	Right half-smile	Lip Pucker	Right cheek puff
Neutral Face	0.00	0.13	0.31	0.31	0.09	0.14	0.19	0.13
Jaw Open	0.74	0.15	0.30	0.16	0.02	0.10	0.27	0.31
Jaw Slide right	0.30	0.35	0.10	0.37	0.01	0.47	0.53	0.10
Left Cheek Puff*	0.01	0.36	0.45	0.16	0.00	0.75	0.81	0.05
Left Eyebrow Raiser	0.18	0.08	0.03	0.61	0.01	0.01	0.43	0.03
Left Half-smile	0.15	0.15	0.09	0.23	0.20	0.01	0.06	0.03
Right half-smile	0.00	0.23	0.09	0.38	0.19	0.97	0.10	0.02
Lip Pucker	0.06	0.18	0.04	0.25	0.23	0.44	0.22	0.02
Right cheek puff*	0.02	0.06	0.02	0.06	0.40	0.01	0.60	0.41

Command ID	Kinect™	Command ID	Kinect™
1	30	5	30
2	30	6*	-
3	30	7	30
4	30	8	30

Eye-tracker™ average error = 344 pixels (Point and Select games)

Eye-tracker™ Mouse Speed

Commands	Seconds Per Command Eye-tracker™	Seconds Per Command Tactile Mouse
30	2.8	1.4

Kinect ™ Face-Tracker Speed

Level	Time (Face-Tracker)	Time (Total)
1	75.38	103.86
2	98.96	102.35
3	114.6	103.81
4	154.59	134.23
5	136.06	114.5
Total	579.59†	558.75†

Conclusions

- Technology to enable quadriplegics to play computer games **must rely on muscles above the neck**
- A large number of popular games can be classified as follows:
  - **Point of View**—8 keyboard commands and a mouse to pan the view
  - **Single input**– Mouse to move to a specific point and a singe command to select
- Devices on the market **do not** meet the **functionality**, **usability**, and **cost** needs of the demographic
  - A **natural user interface** that can recognize facial gestures and track eye movement can be developed using **commercially available technology**
- A **successful** keyboard prototype was developed using Kinectv2 ™
  - Select animation units provided by vision processing application were input to a trained neural network to identify the command to send to the game
  - The system met the speed criteria and accuracy requirements
  - Selecting the appropriate animation units for the user's facial structure is crucial to the success of the device
- A **successful** mouse emulator was developed using an eye tracking device
  - The device was usable for both view panning and point and select applications
  - The accuracy of the mouse emulator was worse than a regular tactile mouse – but was **adequate for game playing**
  - The speed of the mouse emulator was **close** to that of the tactile mouse
- Neural networks enabled **recognition of different variations** of the same facial gesture and hence better predict user intent. It also allows easy **customization** for a specific user

Implications

- Everyday actions can be classified as Unimodal or Multimodal
  - Unimodal: Turning on lights and changing television channels
  - Multimodal: Driving a car (steering and gas/brake pedal) and using the computer
- Most accessibility research has gone into the creation of unimodal systems.
  - However, our most important actions require multiple input channels
- The prototyped interface can be expanded to applications outside of gaming, including surfing the internet and driving a car
  - At its core, the interface recognizes facial gestures and eye movements and outputs them as commands

Future Research

- Exploration of new device customization options
  - Facial structure and user ability determine which Animation Units can be easily recognized
  - For example, cheek puffing in a round face is hard to recognize and users differ in their ability to move different facial muscles.
- Addition of more commands
  - Kinect ™ provides raw 3D coordinates on 1000 facial points, but only 17 Animation Units.
  - Explore possibility of using raw data to enable more commands
- Explore implementation of next-command prediction

References and Acknowledgements

References *(abridged)*:

[1] Brault, Matthew W. *Americans with Disabilities: 2010*. Census Report. U.S. Census Bureau. Washington D.C.: U.S. Department of Commerce: Economics and Statistics Administration, 2012. Document. 22 September 2014.

[2] Cohn, Jeffrey F., et al. "Automated Face Analysis by Feature Point Tracking has high Concurrent Validity with Manual FACS Coding." *Psychophysiology* 36 (1999): 35-42. Print. August 2014.

[3] Jacob, Robert J.K. "Eye Movement-Based Human-Computer Interaction Techniques: Toward Non-Command Interfaces." (n.d.): 1-58. Print. September 2014.

—. "Eye Tracking in Advanced Interface Design." (n.d.): 1-54. Print. August 2014.

Karray, Fakhreddine, et al. "Human-Computer Interaction: Overview on the State of the Art." *International Journal on Smart Sensing and Intelligent Systems* 1.1 (2008): 137-158. Print. August 2014.

[4] Myers, Brad A. "A Brief History of Human Computer Interaction Technology." *ACM interaction* 5.2 (1998): 44-54.

Streets, Graham. *Quadriplegia - A View from the Chair*. n.d. Web. 22 September 2014. <http://www.streetsie.com/quadriplegia/>.

Image Credits (in order of appearance):

[5] Head-tracking Gyroscope and face-tracking mouse—photo taken by student

[6] A sip-and-puff controller—Adaptive Switch Laboratories

[7] Kinect ™ and Kinectv2 ™ sensors—images created by Microsoft

[8] Full interface with Kinect™ and Eyetribe Tracker™—photo taken by student

[9] 9 facial structures—kpophairstyles.tumblr.com

[10] Diagram of a Neuron--biomedicalengineering.yolasite.com

[11] Neuron Action potential graph—wikipedia commons

[12]Neural Network Diagram, equation, notation, graph, and derivision—Ryan Harris

[13] CANDIDE3 wire mesh and animated facial models—courtesy of Microsoft

[14] EyeTribe™ Tracking box—courtesy of the EyeTribe ™

[15]eyeball clipart—clipartpanda.com

[16] Images courtesy of (a)Unbalance, (b)Valve, and (c)Mojang

[17] Image created by student