Access via a Multiple Camera Tongue Switch for Children with Severe Spastic Quadriplegic Cerebral Palsy

by

Brian Robert Pue-Wai Leung

A thesis submitted in conformity with the requirements
for the degree of Master of Applied Science
Graduate Department of Biomaterials and Biomedical Engineering
University of Toronto

Copyright © 2009 by Brian Robert Pue-Wai Leung
Abstract

Access via a Multiple Camera Tongue Switch for Children with Severe Spastic Quadriplegic Cerebral Palsy

Brian Robert Pue-Wai Leung
Master of Applied Science
Graduate Department of Biomaterials and Biomedical Engineering
University of Toronto
2009

Access technologies facilitate novel and alternative methods for individuals with disabilities to interact with their environment. Finding suitable access solutions for children with severe spastic quadriplegic cerebral palsy can be difficult because of their poor motor control and targeting abilities due to spasticity at the limbs, neck, and head. In this research a multiple camera tongue switch was developed for a 7 year-old case study participant with severe spastic quadriplegia. Remotely via video, this system reacts to tongue protrusions as cues for single-switch access. Having multiple cameras mitigates targeting problems with the head that conventional single camera systems would present. Results of a usability experiment with the participant show that good sensitivity (82%) and specificity (80%) can be achieved with a non-contact tongue protrusion access modality for a user with spastic quadriplegia. Moreover, the experiment verified that the extra cameras improve utility of video-based access technologies for the target population.
Acknowledgements

First, I wish to express my gratitude to Dr. Tom Chau for his mentorship. His dedication and passion towards paediatric rehabilitation are simply awe-inspiring. I am privileged to have Dr. Chau as my thesis supervisor.

I would like to thank members of the PRISM lab for offering their thoughts and encouragements and for promoting an exciting research environment.

I take this opportunity to recognize the sources of funding for this research, namely the Natural Sciences and Engineering Research Council of Canada (NSERC), the Bloorview Childrens Hospital Foundation, and the Barbara and Frank Milligan Graduate Fellowship.

I would also like to thank Dr. Allan Jepson (computer vision) and Dr. Denise Reid (rehabilitation sciences) for serving on my program committee and sharing their insights and opinions regarding this research.

Finally, I express my utmost gratitude to my parents, brother, and sister-in-law for their unconditional love and support.
Contents

1 Background 1

1.1 Access technologies ................................................. 1
1.2 Cerebral palsy .......................................................... 4
1.3 Significance of finding solutions for access for children with disabilities . . 6
1.4 Access for children with spastic quadriplegia ....................... 7
  1.4.1 Facial gestures as potential access pathways .................. 7
  1.4.2 Direct contact versus non-contact access technologies ......... 8
  1.4.3 Video-based access technologies ............................... 9

2 Objectives, Research Question, and Rationale 14

3 Case Study Methodology 16

3.1 Participant profile ...................................................... 17
3.2 Justification for and limitation of single descriptive case study ........ 18

4 Multiple Camera Tongue Switch 20

4.1 Multiple cameras as independent input sources ....................... 21
4.2 Example hardware implementation ..................................... 23
  4.2.1 User and system setup .......................................... 23
  4.2.2 Camera and mounting hardware .................................. 25
  4.2.3 Processing unit .................................................. 26
## List of Tables

5.1 Usability experiment: Performance of the multiple camera tongue switch prototype .................................. 71

5.2 Usability experiment: Performance of the center-camera-only tongue switch prototype .................................. 72

5.3 Usability experiment: Comparing the prototype’s performance between full fusion algorithm inputs and center-camera-only inputs ............................... 73
List of Figures

1.1 A conceptual framework of access technologies .......................... 2

4.1 Overview of the multiple camera tongue switch .......................... 20
4.2 High-level architecture of the multiple camera tongue switch ........... 22
4.3 Example hardware deployment .............................................. 23
4.4 User and system setup .................................................... 24
4.5 Example hardware: Linear camera array on a mounting platform ....... 26
4.6 Example hardware: Single-switch output .................................. 27
4.7 Single camera algorithm: An overview ..................................... 28
4.8 Single camera algorithm: Face localization ............................... 30
4.9 HSV colour space .......................................................... 31
4.10 Skin-segmented image for camera configuration ......................... 34
4.11 Face localization example: Face frontal view ............................. 35
4.12 Face localization example: Face side view ............................... 35
4.13 Single camera algorithm: Mouth localization part one .................. 36
4.14 Example of HSV preprocessing ............................................ 38
4.15 A closed mouth example of the colour feature map ..................... 41
4.16 Single camera algorithm: Mouth localization part two .................. 42
4.17 Examples of mouth localization ............................................ 45
4.18 Motion history image example: Left-to-right head displacement ....... 49
4.19 Motion history image example: Right-to-left head rotation ............. 49
Chapter 1

Background

1.1 Access technologies

Able-bodied individuals are constantly interacting with their environment. *Functional intent* refers to the functional state of being that an individual wishes to achieve (e.g. illuminating a dark room). Any task that satisfies functional intent is called a *functional activity* (e.g. turning on the lights of the room). Sometimes the path from functional intent to functional activity is direct and independent (e.g. an individual flicks a switch to turn on the lights of the room). Other times, the path is indirect (e.g. an individual calls for the help of another person to turn on the lights). Interactions between the individual and his or her the environment are necessary in either case, whether they be directed towards a device or another person [45].

For individuals with severe and/or multiple physical disabilities, the path from functional intent to functional activity can be severed even in the most ordinary circumstances because of the inability to interact. As an example, consider a non-verbal individual with no independent mobility in the aforementioned scenario of illuminating a dark room. This individual can neither move beside the light switch to toggle it nor call for someone else to turn on the lights. The individual is left frustrated because his or her functional intent
Access technologies facilitate novel and alternative methods for individuals with disabilities to interact with their environment, in order to bridge the gap between functional intent and functional activity (i.e. enabling access) for these individuals. A generic specification of access technologies is that they sample cues for action from the user and then act on these cues to invoke functional activities. A comprehensive review of access technologies can be found in [45]. Figure 1.1, taken from [45], provides an overview of access technologies with respect to the technology user’s environment.

![Conceptual Framework of Access Technologies](image)

Figure 1.1: A conceptual framework of access technologies.

In this conceptual framework, an access technology consists of an access pathway and a signal processing component. The access pathway is the method of sampling the cues for action. It depends on the nature of the cues and the technology used to detect them. For example, if muscle contractions are the cues for action and they are being detected by surface electromyography sensors, then surface electromyography is the access pathway. The signal processing component hosts the algorithms that process the cues for action. Access technologies are typically connected to the input of another device that actually realizes the functional activity; they enable access because the user cannot operate the
user interface of the other device otherwise.

The simplest output of an access technology is a binary channel that toggles between two states. Because this behaviour mirrors that of a basic mechanical switch, this output modality is commonly known as single-switch output. Despite its simplicity, single-switch output is powerful with regards to enabling access. This is because there exists many software and devices that inherently or can be adapted to support single-switch input. For example, devices for augmentative and alternative communication (AAC), such as the DynaVox V, usually supports single-switch input as a standard feature. To facilitate navigation of the user interface with a single switch, these AAC devices employ a scanning paradigm where the user interface automatically cycles through all its interface elements until the user selects one by a single-switch activation. An access technology that features only one single-switch output is often just called a switch, and such a device is said to enable single-switch access.

The scope of an individual’s abilities and his or her environmental circumstances together determine the viable access pathways for this individual and the utilities of different access technologies. An access technology may work well in one case but not necessarily in another similar situation. Instead, the best solution is the one that best fits the user within his or her personal, environmental, and occupational context [13]. Further, various types of access technologies should be treated as complementary rather than competing solutions to a common challenge. To maximize the options available to prospective users of access technologies, rehabilitation scientists must continue to develop novel access pathways and enhance existing solutions. This is especially relevant to users with severe and/or multiple physical disabilities because the number of feasible access technologies for this user population are very limited at present.
1.2 Cerebral palsy

Cerebral palsy is an umbrella term for a group of neurodevelopmental conditions that affect individuals from early childhood and persist throughout their lives [1]. There was much debate in the past hundred years over the definition and clinical significance of cerebral palsy, because the etiology of cerebral palsy is highly heterogeneous and not well understood [13]. At the International Workshop on Definition and Classification of Cerebral Palsy held in 2004, and led by an Executive Committee for the Definition of Cerebral Palsy, researchers arrived at the following updated definition for cerebral palsy:

“Cerebral palsy describes a group of disorders of the development of movement and posture, causing activity limitation, that are attributed to non-progressive disturbances that occurred in the developing fetal or infant brain. The motor disorders of cerebral palsy are often accompanied by disturbances of sensation, cognition, communication, perception, and/or behaviour, and/or by a seizure disorder.” [1]

The definition above highlights three key characteristics of cerebral palsy. First, people with cerebral palsy have motor impairments. They experience difficulty controlling movements and maintaining postures, to the extent that they have trouble executing activities (activity limitation). Second, cerebral palsy is a discrete developmental disorder of the brain that occurs before, at, or shortly after birth. Third, the disorder is non-progressive — the neurological impairment will not worsen with time [1, 13].

The symptoms of cerebral palsy are also highly heterogeneous. Based on the motor impairments, one can categorize cerebral palsy into the following four types: spastic, ataxic, athetoid, and mixed [13]. Spastic, the most common type accounting for 70%–80% of all cerebral palsy cases, is characterized by muscle stiffness, involuntary spasms, and hypertonia (unusual increase of muscle tone). In contrast, people with ataxic cerebral palsy, which accounts for less than 5% of all cerebral palsy cases, have difficulty con-
trolling their muscles for balance, leading to clumsy movements and postures, tremors, and hypotonia (low muscle tone). Athetoid cerebral palsy accounts for 10%–15% of all cerebral palsy cases and is characterized by involuntary yet continuous writhing movements. People with athetoid cerebral palsy have difficulty maintaining a specific posture. Individuals with symptoms from more than one type are said to be affected by mixed cerebral palsy.

Spastic cerebral palsy is further divided into three sub-types according to the distribution of the affected limbs. About 35%–40% of all spastic cerebral palsy cases belong to spastic hemiplegia, where the spasticity is worse on one side of the body than the other and the arms are usually more affected than the legs. Spastic diplegia cases account for 25%–35% of all spastic cerebral palsy cases. They are characterized by spasticity on both sides of the body, but the legs are more severely affected than the arms. The most prevalent type, accounting for 40%–45% of all spastic cerebral palsy cases, is spastic quadriplegia. It means that all four limbs are equally affected by spasticity and usually the muscles at the mouth, tongue, and pharynx are affected as well. Individuals with spastic quadriplegia are most susceptible to disturbances of sensation, cognition, and/or communication. This is either because the underlying brain lesions that affected neuromotor development disturbed areas of the brain responsible for sensory function and cognition, or because the severe motor impairment prevented typical development via interactions with the environment during childhood [13].

The severity of the symptoms varies widely amongst individuals with cerebral palsy. The Gross Motor Function Classification Scale is a five-level metric for gauging the effects of an individual’s motor impairment on their day-to-day life [32]. For mild cases (i.e. level 1), affected individuals may regain enough muscle control after extensive physiotherapy to engage in day-to-day activities independently. In more severe cases (i.e. levels 4 and 5), self-mobility is limited even with the use of assistive technology [22].
There is no cure for cerebral palsy. Treatment consists of therapies with a physiotherapist, an occupational therapist, a speech and language pathologist, and a psychologist, among other health and allied health professionals. Orthopedic surgery is sometimes warranted to improve the muscle function and conditioning of affected individuals [13, 22].

1.3 Significance of finding solutions for access for children with disabilities

Children can enrich their learning experiences by exploring and manipulating their environment. Real-time interactions with people, the environment, and multimedia are crucial to the development of a child’s cognition and communication skills [26, 37, 40].

A relevant example that highlights the need for access technologies for young children with disabilities can be found in elementary school education. Since the past decade, elementary education programs have balanced passive instruction and active learning so that children may begin their intellectual and creative development through diversified learning perspectives and experiences at an early age [9].

In recent years, the ability to access computers has become essential in order for a child to receive maximal benefit from school education. According to a Statistics Canada survey of 15,500 elementary and secondary schools on the state of integration of information and communications technologies in 2003–04 [35], nearly all schools reported to have used computers for the purposes of “lesson preparation, execution or evaluation during the 2003–04 school year”. The study reported 95% of the 12,100 Canadian elementary schools that responded to the survey offer “educational, drill and practice” software to their students. Furthermore, 33% of the elementary schools reported frequent use of internet/intranet for information retrieval, 30% reported frequent use of “software for special needs students and/or remedial programs providing individualized learning”, 27% reported frequent use of internet for online learning, and 28% reported frequent
use of “software for specific subject areas”. These results confirm that computers have profound impact on modern teaching methods; learning via computers is becoming as important and ubiquitous as traditional blackboard instruction.

Just as the severity of motor impairment varies widely amongst children with cerebral palsy, so do their cognitive and communication abilities. Because the brain lesions do not necessarily affect the parts of the brain responsible for cognition, it should not be a surprise to find children with cerebral palsy possessing typical levels of intellectual potential. Furthermore, it has been documented in literature that “computers can enhance areas such as play, communication, and social interaction in children who are severely disabled, hence aiding childhood development” [18].

Unfortunately, attempts to find a suitable access modality for children with severe cerebral palsy do not always end with success. Because of the heterogeneity of cerebral palsy, success with a particular access solution also varies widely on a case-by-case basis. Without any access technologies prescribed for them, these children cannot use many of the learning tools that typically developing children have regular exposure to. The lack of an access solution has been identified as a primary risk factor for “higher incidence of developmental and learning problems in children with cerebral palsy” [22].

1.4 Access for children with spastic quadriplegia

1.4.1 Facial gestures as potential access pathways

Children with severe spastic quadriplegia have few viable access pathways at their disposal. The use of arms and legs for access is generally not reliable because the limb movements are too sporadic, too inconsistent, and often involuntary. Moreover, their limbs are sometimes restrained to the wheelchair to protect against injuries caused by sudden and involuntary limb movements. For these reasons, individuals with spastic quadriplegia often have difficulty targeting buttons and switches with their arms, legs,
or head. Therefore, mechanical buttons and switches are generally not feasible for individuals with spastic quadriplegia.

On the other hand, facial gestures have potential of being reliable access pathways for children with spastic quadriplegia. After all, children with severe cerebral palsy can communicate their desires, albeit on a limited basis, to experienced caregivers via facial expressions, such as smiling for example [13]. There is a chance that the same facial gestures involved in creating facial expressions can be harnessed as access pathways. Blinking one’s eye(s), raising one’s eyebrow(s), opening and closing one’s mouth, and protruding one’s tongue are some examples of facial gestures that may be used for access [2, 16].

1.4.2 Direct contact versus non-contact access technologies

One can circumvent the problem of targeting mechanical switches and buttons by attaching input sensors directly onto the user, thus giving rise to the class of direct contact access technologies. There are many published works in human-computer interaction (HCI) and rehabilitation engineering literature on direct contact technologies driven by head or facial movements. For example, the EagleEyes [14] and a system proposed by Hori et al. [19] use electro-oculograms to build single-switch outputs by measuring electrical activity from eye movements with electrodes placed around an eye. Struijk developed a system that constructs a single-switch output from tongue pressure, measured using a pressure sensor mounted on a dental accessory worn inside the mouth [43]. Chen et al. proposed an infrared transmitter attached on eyeglasses to operate an electronic keyboard panel that is mounted beside the computer monitor and facing the user [6].

There are three disadvantages pertaining to direct contact access technologies for children. First, usability of the input sensors may degrade over time. Switches and positioning hardware may be displaced by sudden, inadvertent head or limb movements, necessitating frequent manual rearrangement. Similarly, for direct contact modalities
which harness surface biopotentials, the conductivity of the electrode-skin interface may become compromised in the presence of perspiration [2]. Second, children tend to dislike having an array of sensors placed on their bodies [2]. Third, children may also feel uneasy wearing extra accessories that accentuate their differences [16].

Having considered the usability issues of ordinary mechanical switches and direct contact access technologies, non-contact access technologies would seem ideal in term of best accommodating the physical abilities of individuals with spastic quadriplegia. Non-contact (or remote) modalities enable access without the need for physical connection to the user. Sensors are still necessary to monitor cues for action from the user, but they are always placed at a distance from the user. Hence, non-contact alternatives overcome the aforementioned limitations of direct contact modalities. Moreover, the targeting problem of ordinary mechanical switches is irrelevant to non-contact modalities. On the other hand, the lack of physical contact creates many technical challenges in the design of non-contact access technologies. It is non-trivial to achieve robust remote tracking and interpretation of deliberate user actions. From the user’s standpoint, the absence of physical contact means there is no tactile feedback accompanying the interactions.

Voice is an obvious pathway for non-contact access. However, individuals with spastic quadriplegia experience spasticity of the muscles at the mouth, tongue, and pharynx — conditions that usually result in problems with speech production and articulation [13]. Therefore, voice by itself is usually not a suitable access pathway for users with spastic quadriplegia.

1.4.3 Video-based access technologies

A video camera as a pathway for non-contact access has garnered much interest in the past decade thanks to the availability of low-cost imaging hardware and substantial increases in the computing power of personal computers and embedded systems [33]. The premise of video-based access technologies is to use video cameras for remote detection of
user movements. A user operates this technology by facing the camera and then performing movements that the system expects as valid input according to an input vocabulary and protocol. Video-based access technologies that exploit facial gestures should be appropriate for users with spastic quadriplegia, because the same head or facial movements that are exploited in some direct contact modalities can also drive a video-based access technology, but potentially with less demand for movement reliability. In practice, some individuals with cerebral palsy find video-based access technologies difficult to use. They find themselves lacking the necessary gross motor control to keep their heads directly facing the camera [2]. At other times, the system has trouble discerning intentional movements from tremors and involuntary reflexes, which can be sudden and unpredictable, and thus it cannot correctly interpret the user’s actions.

Note that access technology research should not be confused with HCI research for able-bodied people. First, one must explicitly design for the physiological condition of the target population\(^1\), whereas the group of able-bodied individuals exhibits a relatively uniform motor physiology. In the case of spastic quadriplegia, the types of user action are often limited to head and facial movements. Second, it is not uncommon that proposed solutions for users with severe motor impairment emulate the functions of a computer mouse or keyboard to facilitate basic human-computer interactions; whereas a main goal of HCI research for able-bodied people is to digress from the keyboard and mouse paradigm and towards more natural methods of human-computer interaction, such as haptic and gesture interfaces [39].

One family of video-based access technologies that has been tried on users with severe motor impairment is eye gaze systems. The basic premise of eye gaze systems is to exploit horizontal and vertical eye movements, which can be deduced from changes to

\(^1\)For example, an individual with C1-C2 spinal cord injury and another individual with severe cerebral palsy are both considered to have severe motor impairment. Yet, the former usually has little exploitable movements but can control these residual movements well, whereas the latter potentially has lots of movements that are poorly coordinated, imprecise, and accompanied by involuntary reflexes.
the eye gaze direction, as two single-switch outputs or as direction cues for 2D navigation. Advanced use of eye gaze control involves estimating the direction of eye gaze to infer the user’s focus of attention on the computer screen. This enables a user to select from several icons displayed on the computer screen just by looking directly at the icon [23]. The main challenge with eye gaze systems is robust tracking of the iris pose. Yoo et al. [49] proposed a solution that uses a near-infrared CCD camera to detect four corneal reflections shone by four infrared LEDs mounted at the corners of a computer screen. The four-sided polygon formed by the four corneal reflections is treated as a projection of the computer screen. The pupil is located from a second image of the eye illuminated by a fifth infrared LED. The pupil’s position relative to the inside of the polygon infers the user’s focus of attention on the screen. Itoh and Ifukube [23] proposed an eye gaze system for individuals with severe limb disability to operate an onscreen Japanese keyboard. The gaze direction is estimated from the position of the pupil center relative to a corneal reflection point. The darkest spot and brightest spot of an infrared corneal reflection determine the pupil center and corneal reflection point respectively. A foreseeable problem with the eye gaze pathway is that the eyes can be easily fatigued from frequent voluntary eye movements. This hinders the utility of eye gaze systems during long sessions of use [7].

A second group of video-based access technologies exploits head movements to control a computer mouse. The two main design challenges to these systems lie with robust tracking of the physical movements for mouse cursor control and the algorithm for translating the physical movements into mouse cursor movements. The Camera Mouse by Betke et al. [2] is a single camera system that achieves feature generic non-contact mouse control. Generic feature tracking is realized by sampling a template sub-image of a user feature specified manually with the help of an operator, usually the caregiver. Tracking is done by finding the best correlation between the current video frame and the template sub-image centered at pixels within a small neighbourhood of the previous feature coordinates. A remarkable component of the study is that the Camera Mouse was tested
with individuals with cerebral palsy. Although no quantitative performance results were published, the authors reported that the system was suitable for and well accepted by most test subjects, and some subjects have even continued to use the system on a daily basis.

The Head Mouse by Su et al. [44] also uses the template matching technique for tracking the eyes. The proposed algorithm implements automatic eye detection and therefore provisions for lost tracking. The key contribution is a relative cursor movement algorithm where mouse cursor movements depend on the displacement direction, rather than magnitude, of the feature coordinates from time $t$ to $t + 1$. Moreover, each cursor movement is scaled by a dynamic continuous gain factor that speeds up movement if the displacement direction is held for longer durations. The FaceMouse by Perini et al. [34] considered a simpler relative cursor movement algorithm that uses two discrete velocity values in each of the horizontal and vertical directions. The velocity selection depends on the displacement of nose with respect to the origin of a $3 \times 3$ movement direction grid. One more example is a system by Kjeldsen [25] that relies on head tilting for mouse cursor control and uses a sigmoidal transfer function to smooth out the cursor movements.

There are relatively few published works on video-based access technologies that harness facial gestures. The best examples are the BlinkLink and EyebrowClicker systems proposed by Grauman et al. [16]. Both are single camera systems that look for voluntary eye blinks or eyebrow raises respectively as cues for computer mouse clicks. Both systems first locate the eyes automatically, using several stages of image binarization, morphological erosion, and anthropomorphic filtering on a few initial input video frames to isolate the eyes. For BlinkLink, the system saves a sub-image template of the open eye from known anthropomorphic properties about the open eye. Finally, it detects a closed eye on the premise that a closed eye does not correlate well against an open eye template, causing the normalized correlation coefficient to drop below some threshold. An intentional blink is distinguished from an involuntary blink by the duration of the eye
blink. The EyebrowClicker algorithm uses a similar idea of thresholding on the eyebrow height to trigger mouse clicks. The authors reported detection accuracies of 95.6% for BlinkLink and 89.0% for EyebrowClicker based on testing with 15 subjects. However, it was not stated whether the test subjects were able-bodied or have motor impairment.

Despite recent advances in the design of video-based access technologies for individuals with severe motor impairment, they still have limited utility for individuals with severe spastic quadriplegia due to the prevalence of extraneous and involuntary muscle activities. In particular, a single camera system presents yet another targeting problem for these users — the problem of maintaining the gesture source (e.g. the face) inside the camera’s field of view. At this time, there does not appear to be any published works in assistive technology literature that address this problem.
Chapter 2

Objectives, Research Question, and Rationale

The objectives of this research in access technologies were:

1. to design a switch activation algorithm for children with spastic quadriplegia that uses input video data from multiple cameras as independent information sources to maximize sensitivity and specificity of intentional facial gestures, and

2. to quantify performance of the multiple camera system as an access solution in a case study design with a child with spastic quadriplegia, through caregiver and participant feedback and performance logging, in the context of a single-switch selection task.

These led to the following research question:

Can information from multiple independent camera channels enhance the detection of intentional facial gestures?

The concept of using multiple cameras in a video-based HCI system was not new. Various HCI research have proposed the use of two cameras, either in stereo-vision or
by triangulation, to realize robust pose estimation of a user’s hands or head for applications in real-time 3D modelling and 3D gesture recognition [24, 50]. In this research, the emphasis was on the design of a multiple camera access technology and the investigation of whether multiple cameras can enhance the performance of video-based access technologies for children with spastic quadriplegia.

The motivation for investigating multiple cameras was raised by empirical videos, which showed that extra video perspectives could potentially extend coverage of the frontal face view of a user with severe spastic quadriplegia, for longer durations. This is significant because typical video-based facial gesture recognition systems work best with videos that have the frontal view of the user’s face. The empirical videos were collected during an informal facial gestures assessment session with a child with severe spastic quadriplegia.\(^1\) The recording setup consisted of two Logitech Quickcam Pro 4000 webcams powered by separate laptops. One was recording directly in front of the child, while the other was simultaneously recording from the left at a 45° angle. The child was then asked to repeat several facial gestures. In 18 minutes of footage, the frontal view of the child’s face exited the center camera (i.e. rotated to the left or right by 30° or more) for an approximate total of 6\(\frac{1}{2}\) minutes. The bouts of frontal view lost to the center camera persisted for about 30 seconds on average; the shortest loss lasted 14 seconds and the longest one lasted 50 seconds. During moments where the head rotated to the left, the left camera did capture a better frontal view than the center camera. In these cases, video from the left camera would have been more suitable for detecting intentional facial gestures.

\(^1\)This child eventually became the participant for our single case study.
Chapter 3

Case Study Methodology

Development of the multiple camera access technology was conducted within the context of a single descriptive case study. A participant (henceforth addressed as “the participant”) was recruited prior to the commencement of prototype design and implementation. We were approached by the parents of the participant to find an access solution for their child. In particular, the parents were interested in a non-contact access solution because of the participant’s difficulty with targeting mechanical switches and safety concerns for his spastic movements coming in contact with wheelchair-mounted devices. The participant completed an initial assessment for potential access pathways, which identified tongue protrusion as a facial gesture that could be harnessed for single-switch access. Concurrent to the preparation of a prototype, the participant underwent training with teachers and caregivers to strengthen his control of tongue protrusions. Finally, the participant was brought back for repeated sessions of a usability experiment to evaluate performance of the prototype. The intention in following this process was to encourage direct feedback from a potential future user of the multiple camera access technology early in the design process, with the expectation of improving adoption and success of the resulting device [18, 29].
Chapter 3. Case Study Methodology

3.1 Participant profile

The participant was a 7 year-old Eurasian boy. He had spastic quadriplegia with severity measured to the most severe grade of level 5 on the Gross Motor Function Classification Scale (see section 1.2). In particular, he had no independent mobility and sat in a manual wheelchair while in the community. His four limbs and head were affected by sporadic yet frequent bouts of spasticity. At times, the spastic muscle activity would persist for an extended period (e.g. more than 30 seconds) and thus rendered him stuck to a pose. Due to this lack of adequate voluntary control of his limb and head movements, he was unable to achieve consistency in targeting mechanical switches located proximal to his body.

Tongue protrusion was identified as a potential movement for single-switch access for this participant. The participant had voluntary control of his facial movements. He was able to smile and express displeasure. He struggled at performing voluntary eye blinks with consistency and had even greater difficulty with voluntary eyebrow movements. He had adequate control of his lips, although the quality of control diminished in the presence of head and neck spasticity. The participant was able to produce voluntary tongue protrusions with consistency. During spasticity, he would be too distracted by discomfort to attempt tongue protrusions. Furthermore, the participant rarely produced involuntary tongue protrusions. Hence, tongue protrusion was selected for single-switch access on the basis of being the facial gesture that could be most reliably modulated by the participant.

The participant was capable of recognizing pictures and words. He had past experience working with computers, completing multiple choice exercises on a classroom computer with caregiver assistance. The participant could comprehend verbal instructions and react to verbal inquiries. He was considered non-verbal but could voluntarily articulate one-word responses to indicate preference or YES/NO decisions. The participant was admitted to this study on the premise of these inclusion criteria.
The participant had yet to be fitted with an access technology when he entered into the study. His past history of switch-fitting included an assortment of wheelchair-mounted mechanical switches, sip-and-puff switch, and voice-activated switches. Instead, he relied on caregiver-mediated access, i.e. a caregiver physically issuing commands on his behalf. Caregiver-mediated access was useful to facilitate tongue protrusion training — by instructing the caregiver to act if and only if the participant performed a tongue protrusion. The participant received three months of tongue protrusion training, averaging two 30-minute sessions per week, with the aforementioned caregiver-mediated access regime. The participant was repeatedly reminded that tongue protrusions are only appropriate during computer activity.

3.2 Justification for and limitation of single descriptive case study

The decision to pursue only one case study was due to the challenges of recruiting multiple participants with similar characteristics and abilities. Ideally, all participants should test the same prototype with the same set of activities. However, only one other candidate among the four respondents to our call for participants had consistent tongue protrusions. Moreover, this candidate was less cognitively capable and less responsive to visual stimuli such as pictures and words. This discrepancy in sensory and cognitive abilities between the candidates would confound comparisons of the two cases with regards to performance of the prototype. The difficulty in recruitment was not unexpected because symptoms and functional abilities vary widely amongst individuals with severe spastic quadriplegia [13].

Single-subject research designs were not applicable to this study because an appropriate baseline was not available. Single-subject designs statistically quantify the effects of one or more treatments by implementing a rigorous schedule of treatments over the course of the study, thereby increasing the level of evidence despite a sample size of one [38].
However, caregiver-mediated access is not a meaningful baseline. On one hand, all access technologies are trivially superior because caregiver-mediated access does not support independent access. On the other hand, they are trivially inferior because caregiver-mediated access achieves perfect performance, as long as the caregiver is able to observe the cues for access.

It should be noted that a reduction of the level of evidence is the compromise for implementing a single descriptive case study. Specifically, descriptive case studies provide only anecdotal evidences and anecdotal insights; they are not rigorous enough for statistical inferencing [38].
Chapter 4

Multiple Camera Tongue Switch

As a device for facilitating single-switch access, a multiple camera tongue switch maps intentional tongue protrusions from the human user into switch activations. Figure 4.1 shows the black box representation of the tongue switch arranged as an instance of the access technology conceptual framework (see section 1.1). Cameras are the access pathway because their video streams capture the tongue protrusions. Signal processing consists of colour video processing of the video inputs from the multiple cameras.

![Diagram of Multiple Camera Tongue Switch](image)

Figure 4.1: Multiple camera tongue switch as an instance of the access technology conceptual framework.
4.1 Multiple cameras as independent input sources

The cameras of a multiple camera computer vision system may be tightly coupled to each other or totally independent. In a tightly coupled configuration, each camera’s pose must be calibrated relative to the others. The computer vision system maintains knowledge of the cameras’ poses and the accuracy of this information is essential to correct run-time behaviour. Computer stereo-vision [10] is an example of a tightly coupled computer vision system. Conversely, knowledge of the cameras’ poses is unnecessary when the cameras are totally independent. Therefore, cameras may be repositioned, added, or removed at run-time without compromising correct behaviour (although system effectiveness may change due to a different set of viewpoints to the external environment). As an example, simple video surveillance systems are typically set up with cameras in an independent configuration.

The multiple camera tongue switch implements an array of independent cameras. Hence, this multiple camera setup is functionally analogous to deploying several instances of a single camera tongue switch observing the user from distinct perspectives. The high-level system diagram shown in figure 4.2 summarizes this interpretation of the multiple camera system.

For each of the $N$ cameras, an instance of the same single camera algorithm performs dedicated video processing for that camera and only on the video input supplied by that camera. Thus, every “camera and single camera algorithm instance” pair is a complete instance of the same single camera system. Data from the multiple cameras are pooled together only at a later processing stage. A fusion algorithm samples the results from the $N$ single camera algorithm instances in order to calculate a single statistic that will determine whether a switch activation should be made. The multiple camera fusion algorithm does not process any raw input from the $N$ cameras. Likewise, none of the $N$ single camera instances have direct control of the switch output. Cameras may be freely added or removed as long as 1) each camera is handled by a dedicated instance of the
Figure 4.2: High-level system architecture of the multiple camera tongue switch. The tongue switch features two stages of processing. Note that data flow is directed upwards.

single camera algorithm, 2) compatible data exchange exists between each single camera algorithm instance and the fusion algorithm, and 3) the fusion algorithm is dependent only on the number of cameras and has generic treatment of the cameras otherwise.

The aforementioned flexibility with repositioning, adding, and removing cameras in an independent configuration provides desirable clinical benefits as well. The system is simpler for a caregiver to set up by virtue of not requiring calibration of the cameras’ poses. Moreover, the system can still function properly if the caregiver or the user accidentally alters the cameras’ poses via physical contact. Finally, on a case-by-case basis the caregiver may add more cameras to extend coverage of the user’s face in the
presence of spastic head movements. In practice, computer processing resources and camera mounting problems limit the number of cameras that can be feasibly deployed.

### 4.2 Example hardware implementation

Hardware implementation of the multiple camera tongue switch consists of a camera array, a processing unit (typically a computer), and an electronic circuit to deliver single-switch output. Figure 4.3 demonstrates the hardware setup for the example application of single-switch access to a desktop computer.

![Diagram of Hardware Setup](image)

Figure 4.3: Example hardware deployment of the multiple camera tongue switch. The single-switch output of the tongue switch clicks the left button of a modified computer mouse. Thus, the tongue switch facilitates single-switch access to the desktop computer.

Many implementations of the tongue switch are possible by changing the number of cameras, the types of cameras used, and the computer configuration. The following subsections describe the prototype implementation that was developed for the case study.

#### 4.2.1 User and system setup

The prototype implementation consists of three cameras in a linear arrangement and at a distance from the user. Figure 4.4 shows, from a top-down perspective, the positioning
of the linear camera array relative to the user. Similar to a single camera setup, the user is aligned with the center camera and positioned at \(x\) cm away. Two peripheral cameras are placed to the left and right of the center camera at \(x\) cm apart, such that the camera-array axis is perpendicular to the user-camera axis. The peripheral cameras are toed-in towards the user so that they monitor the user from the sides at a 45° angle. Suitable values for \(x\) depend on technical factors such as the cameras’ optics. They also depend on environmental factors such as the size of the flat surface that supports the camera array and the dimensions of the user’s wheelchair. Values within the range of 30–70 cm are typical.

![Diagram](image.png)

Figure 4.4: Positioning of the linear camera array relative to the user, from an overhead perspective. The user is a good reference point because he or she is in a wheelchair that is locked stationary.

In this setup the peripheral cameras are farther away from the user than the center
camera. Thus, it is necessary to adjust the zoom of the peripheral camera, either optically or via software, so that all three cameras capture the user’s face at similar scales for more consistent signal processing across cameras. The scaling issue can be mitigated using a circular user-system topology, with the user positioned at the centroid and the cameras forming an arc on the circle of $x$ cm radius. The linear configuration is chosen for the ease of deployment on conventional tabletops.

Also, this setup extends coverage of the user’s face for left and right head movements from spastic activity but not for up and down head tilts. The caregiver can compensate for head tilts by adjusting the height of the camera array and, if possible, adjusting the tilt angle of the user’s wheelchair. Otherwise, provisions for head tilts depends primarily on the angle of view of the cameras.

In practice, the caregiver is not expected to follow these positioning specifications precisely. Instead, the caregiver may position the user in front of the tongue switch by intuition as long as the general topology in figure 4.4 is realized.

### 4.2.2 Camera and mounting hardware

The prototype’s camera array consists of three Logitech QuickCam Pro 5000 webcams mounted on a custom-made platform, as shown in figure 4.5a. Each webcam captures video input at a resolution of $320 \times 240$ pixels and at a frame rate of 15 Hz. High frame rate takes priority in order to facilitate timely handling of intentional tongue protrusions. The resolution is subsequently set as large as possible while sustaining 15 Hz/camera processing with the available computing resources. The three cameras interface with the processing unit via universal serial bus (USB).

The mounting platform is a single plastic board with metal railings for structural reinforcement, rubber feet to reduce slipping, and three ball-jointed monopods for the webcams. The center monopod is fixed to the middle. The peripheral monopods can be repositioned via five mounting holes (figure 4.5b) on both sides, starting at 30 cm and
Figure 4.5: Three Logitech QuickCam Pro 5000 webcams mounted on a custom-made platform in a linear array.

up to 70 cm away from the center camera at 10 cm increments.

4.2.3 Processing unit

The prototype’s processing unit is a laptop computer equipped with an Intel Core 2 Duo T7400 2.16 GHz processor and 2 GB of 667 MHz DDR2 memory. This is sufficient for processing the three video streams at the aforementioned resolution and frame rate. Also, empirical evidence showed that processor throughput is the predominant limiting factor when scaling the multiple camera system with more cameras.

The laptop computer is equipped with three USB ports for the three webcams. These ports must be driven by separate USB host controllers, otherwise there will be data transmission complications due to bus saturation. Omitting all data transmission overheads, streaming of RGB colour video at 24 bits/pixel (8 bits per colour channel), $320 \times 240$ pixels/frame, and 15 frame/second demands a throughput of 276.48 Mbits/second. Host controllers supporting USB 2.0 specification have theoretical capacity of 480 Mbits/seconds. Therefore, it is impossible for a single USB host controller to multiplex two or more such
video streams without dropping frames. The laptop computer from the prototype has two USB host controllers driving two built-in USB ports. A PCMCIA expansion card provides the third USB port and host controller.

### 4.2.4 Single-switch output

For single-switch access via hardware, switch activation corresponds to the delivery of a digital pulse to the target single-switch device over a single wire. The prototype supports hardware single-switch access using a RS-232 serial port, specifically by toggling between logical HIGH and LOW voltages over one of the RS-232 signals (e.g. the data-terminal-ready signal). A USB-to-serial adapter (figure 4.6a), exposing a DE-9 terminal, provides a RS-232 serial port for the laptop computer. Furthermore, single-switch devices typically feature 1/4-inch mono jacks as their electrical interface. A simple adapter (figures 4.6b and 4.6c) bridges the DE-9 terminal to the 1/4-inch mono jack, specifically by binding a 1/4-inch mono plug to the DE-9 pin that carries the toggled RS-232 signal.

![USB-to-serial adapter](image1.png) ![Serial-to-mono adapter](image2.png) ![DE-9 terminal and mono plug](image3.png)

(a) USB-to-serial adapter  (b) Serial-to-mono adapter  (c) DE-9 terminal and mono plug

Figure 4.6: The prototype’s hardware for implementing single-switch output.

It may not be necessary to deploy the hardware above for access to software applications. If there is enough processing power to execute the target software and the multiple camera tongue switch on the same computer, then software output (e.g. software emulated mouse click) can substitute for hardware single-switch access.
4.3 Single camera algorithm

The single camera algorithm realizes a single camera tongue switch. It processes every input video frame for a statistic that can be modulated with tongue protrusions. This statistic is then used as a decision criterion, by thresholding, for triggering switch activations. An overview of the algorithm is shown in figure 4.7.

The algorithm accepts RGB colour video frames as its input. Each original input frame $O_{RGB}^{(t)}$ passes through three stages of processing. First, the face localization module locates the user’s face within the frame. A bounding rectangle $BR_f^{(t)}$ defines the sub-image of $O_{RGB}^{(t)}$ that contains the user’s face, while pixels outside of the rectangle are discarded from further processing. The retained sub-image, forming the face region of interest, passes through the mouth localization module in an attempt to find the user’s mouth. Similarly, a bounding rectangle $BR_m^{(t)}$ defines the mouth region of interest, which is inside the face region of interest, if the mouth is sufficiently visible in the current frame. Finally, the tongue switch module processes both the face and mouth regions of interest. It computes the aforementioned tongue-modulated statistics, denoted by $z_{TMS}^{(t)}$. It also produces a measure $z_{FVM}^{(t)}$ that determines whether the camera is perceiving a frontal view of the user’s face. Also shown in figure 4.7 is a motion-based inhibitor. This module attempts to detect spastic head movements by measuring motion in the face region of interest. It temporarily disables the switch output whenever it measures large amounts of motion.

Figure 4.7: An overview of the single camera algorithm.
Signal source localization are characteristic of non-contact access technology [2, 16]. While it may be difficult to place sensors onto children, localization of the signal source is implicit and trivial once a sensor is properly attached and positioned. Conversely, a non-contact solution mitigates sensor mounting issues, but then correct operation of the solution depends on proper signal source localization.

The algorithm works mainly with colour features. Fundamentals of colour image processing are covered in [15, 36, 41]. In literature, colour features from various colour spaces have been investigated for applications such as face detection [42] and face features extraction [5]. Colour features are of interest because they are usually fast and inexpensive to calculate with regards to computational resources. This is especially relevant to the multiple camera tongue switch because of the multiple instances of the single camera algorithm. More importantly, colour features are used in this algorithm because they are pose invariant and resistant to motion. These two properties are particularly useful in the presence of spastic head movements. Pose invariance means the magnitudes of the colour features are unaffected by deviations in head pose, which are likely after bouts of head spasticity. Resistance to motion means the algorithm is less likely to lose track of the colour features during spastic head movements. In natural images, similarly coloured pixels typically cluster into blobs with low spatial frequencies [17]. Therefore, colour features will likely survive head motion blur, which is an optical low-pass filtering process.

Colour features are not without their disadvantages. Colour features can be confused by illumination of the environment, especially coloured light sources and shadows. Also, proper camera configuration is crucial. Overexposure and underexposure can wash out colour information, and improper white balancing can corrupt the colour features.

Edge (e.g. Sobel, Laplace, and Canny) and corner (e.g. Harris) features were considered for mouth localization. They were not retained for two reasons. First, these structural features typically contain energy at high spatial frequencies [10, 41] and thus
Chapter 4. Multiple Camera Tongue Switch

they can be suppressed by motion. Second, shadows on the face can cause false edges and corners that do not correspond to any face features (i.e. eyes, mouth, etc.).

4.3.1 Face localization

Assume the following preconditions regarding the user and system setup:

- The user’s head is inside each camera’s field of view.
- The patch of skin nearest to the field of view centroid comes from the user’s head.

Then, face localization can be realized using a colour tracking algorithm that is tuned to track skin colour. Figure 4.8 summarizes the face localization module of the single camera algorithm.

![Figure 4.8: The face localization module.](image)

**CAMSHIFT colour tracking**

The face localization module makes use of the CAMSHIFT (continuously adaptive mean-shift) colour tracking algorithm by Bradski [3]. The mean-shift algorithm itself is a non-parametric clustering technique by iteratively ascending the gradient to locate a dominant cluster’s center of mass within a search window [8].

CAMSHIFT works with the HSV (hue, saturation, and value) colour space, specifically the hue component which corresponds to the colour of monochromatic light (see...
Prior to hue (colour) tracking, the algorithm needs a colour model. This is typically obtained by sampling the hue values from the pixels of the object to be tracked, in order to construct an empirical hue histogram $H(h \in \{0, 1, \ldots, 359\})$. It is better to group neighbouring hue values together so that perceptually similar colours share the same bin. However, a histogram with too few bins would limit the number of distinct trackable colours. Normalizing this histogram by $\max(H)$ yields a look-up table $L(h) \in [0, 1]$ — a function that assigns to each hue value a probability that the hue comes from the tracked object.

Figure 4.9: HSV colour space in a cylindrical representation. Hue corresponds to the angular value on a colour wheel. The radial component of the colour wheel encodes saturation. Smaller saturation implies a mixture of many monochromatic colours. The cylinder height encodes value. Value implies the amount of illumination on the colour mixture.

Let $\text{BR}_f^{(t)}$ represent the bounding rectangle of the face region of interest, defined relative to $O_{\text{RGB}}^{(t)}$. For now, suppose that $W_{\text{RGB}}^{(t)}$ is the input to CAMSHIFT. Applying CAMSHIFT on $W_{\text{RGB}}^{(t)}$ yields $\text{BR}_f^{(t)}$, which may be different in position and dimensions to $\text{BR}_f^{(t-1)}$. According to [3], the steps of the CAMSHIFT algorithm are as follows:

1. Convert the input frame from RGB to HSV colour space, i.e. $W_{\text{RGB}}^{(t)} \rightarrow W_{\text{HSV}}^{(t)}$. 

![HSV colour space in a cylindrical representation](image)
2. Extract the hue component $W_h^{(t)}$. Mask away pixels with very low saturation or very low brightness, i.e. set $W_h^{(t)}[m, n] = 0$ for all $\{m, n \mid 0 \leq W_s^{(t)}[m, n] < 25 \text{ or } 0 \leq W_v^{(t)}[m, n] < 25\}$. Note that $W_s^{(t)}[m, n] \in \{0, 1, \ldots, 255\}$ and $W_v^{(t)}[m, n] \in \{0, 1, \ldots, 255\}$ for all $[m, n]$. Pixels that fit these criteria are nearly achromatic and thus the hue values are unreliable (see figure 4.9).

3. Using the look-up table $L(h)$, compute the back projection image

$$W_{\text{projection}}^{(t)}[m, n] = L(W_h^{(t)}[m, n]). \quad (4.1)$$

Indeed, $W_{\text{projection}}^{(t)}$ is a spatial map of probabilities, and the tracked object should reside in a cluster of high probabilities due to hue matching.

4. Apply the mean-shift algorithm on $W_{\text{projection}}^{(t)}$, using $BR_f^{(t-1)}$ as the search window. As long as the tracked object is not moving at very high speeds and the video frame rate is not too slow, a part of the tracked object is usually contained in $BR_f^{(t-1)}$ and so this choice of search window is valid.

5. The new bounding rectangle $BR_f^{(t)}$ centers at the convergence point (center of mass) of the mean-shift algorithm.

The CAMSHIFT algorithm differs from the mean-shift algorithm in the automatic resizing of the search window. The search window enlarges and shrinks according to the tracked object’s area. Thus, the search window is usually large enough to track the object while not too large to include distractors (non-target objects of similar colour) and other noises [3].

The initial search window $BR_f^{(0)}$ is centered at the field of view centroid with the width and height set at 15% of the input frame’s width and height respectively. This choice follows from the precondition that the user’s head is centered at each camera’s field of view. Furthermore, the number of hue histogram bins, the convergence criterion, and the maximum number of iterations for the mean-shift are three parameters of the
CAMSHIFT algorithm. The prototype uses 18 histogram bins, convergence criterion of 1 (i.e. convergence if the candidate center of mass changes by less than one pixel after an iteration), and 10 mean-shift iterations maximum.

**Skin colour segmentation on YCg’Cr’ colour space**

For face localization, the CAMSHIFT algorithm needs to build a skin colour model. To provide guidance for skin colour tracking, the original frame \( O(t) \) is preprocessed with skin colour segmentation before it is fed into CAMSHIFT. Skin colour segmentation is more appropriately done on colour spaces that separate luminance and chromanance \([46]\), such as HSV \([48]\) and YCbCr \([21]\). The face localization module uses skin colour segmentation (see figure 4.8) on the YCg’Cr’ colour space as proposed by de Dios \([12]\). This colour space is similar to YCbCr except for the green colour difference as the second chromatic component.

The first step is to convert \( O(t) \) to the YCgCr colour space via the linear transformation

\[
\begin{bmatrix}
W_Y^{(t)}[m,n] \\
W_{Cg}^{(t)}[m,n] \\
W_{Cr}^{(t)}[m,n]
\end{bmatrix}
= \begin{bmatrix}
16 & 65.481 & 128.553 & 24.966 \\
128 & -81.085 & 112 & -30.915 \\
128 & 112 & -93.768 & -18.214
\end{bmatrix}
\begin{bmatrix}
\frac{O^{(t)}[m,n]}{255} \\
\frac{O^{(t)}[m,n]}{255} \\
\frac{O^{(t)}[m,n]}{255}
\end{bmatrix},
\]

which is defined specifically for 8-bit representations of \( O_R^{(t)}, O_G^{(t)}, \) and \( O_B^{(t)} \). The luma component \( W_Y^{(t)}[m,n] \) is discarded because skin colour is mostly a chromatic feature. A second linear transformation,

\[
\begin{bmatrix}
W_{Cg}'^{(t)}[m,n] \\
W_{Cr}'^{(t)}[m,n]
\end{bmatrix}
= \begin{bmatrix}
-48 & \cos 30^\circ & \sin 30^\circ \\
80 & -\sin 30^\circ & \cos 30^\circ
\end{bmatrix}
\begin{bmatrix}
W_{Cg}^{(t)}[m,n] \\
W_{Cr}^{(t)}[m,n]
\end{bmatrix},
\]

rotates the Cg-Cr plane 30° clockwise to yield the Cg’-Cr’ plane. It was found that skin colour segmentation on the Cg’-Cr’ plane is superior to segmentation on the Cg-Cr plane \([12]\). Finally, the skin-segmented frame \( S_{RGB}^{(t)} \) is obtained by binary thresholding of
the Cg’ and Cr’ components, i.e.

\[
S_{\text{RGB}}^{(t)}[m, n] = \begin{cases} 
O_{\text{RGB}}^{(t)}[m, n], & \text{if } 125 \leq W_{\text{Cg}}^{(t)}[m, n] \leq 140 \text{ and } 136 \leq W_{\text{Cr}}^{(t)}[m, n] \leq 217 \\
\vec{0}, & \text{otherwise.} 
\end{cases}
\]

(4.4)

The CAMSHIFT algorithm receives \(S_{\text{RGB}}^{(t)}\) for skin colour tracking instead of the original input frame \(O_{\text{RGB}}^{(t)}\). Furthermore, the CAMSHIFT algorithm samples the hue values from the non-zero pixels of \(S_{\text{RGB}}^{(t)}\) to construct the skin colour model.

On a side note, the skin-segmented frame \(S_{\text{RGB}}^{(t)}\) is useful during camera configuration. Specifically, the camera exposure and white balance should be adjusted until the retained pixels in \(S_{\text{RGB}}^{(t)}\) actually correspond to skin, especially the user’s head, and other skin-coloured objects. Non-skin-coloured objects should be filtered out as much as possible in \(S_{\text{RGB}}^{(t)}\) (see figure 4.10).

![Sample scene](image1)
![Proper calibration](image2)
![Improper calibration](image3)

(a) Sample scene (b) Proper calibration (c) Improper calibration

Figure 4.10: Using the skin-segmented image for calibrating camera exposure and white balance. When the skin-segmented image retains only skin and actually skin-coloured objects, the camera is properly configured. For the improper calibration shown, the skin-segmented image retains objects that are not skin-coloured, such as the green shirt.

**Sample output**

Figures 4.11 and 4.12 demonstrate face localization for two example scenarios.
4.3.2 Mouth localization

In typical human faces, the lips and tongue are more saturated with red colour than other face features (e.g. eyes, nose, skin, etc.) [28]. The mouth localization module uses a colour-based algorithm that depends on this observation. Specifically, the mouth localization algorithm calculates two colour features that accentuate regions of the face with significant red colour saturation.

The input to the mouth localization module is a sub-image of the original frame $O_{\text{RGB}}^{(t)}$ containing only the face region of interest as defined by the bounding rectangle $\text{BR}_f^{(t)}$. 
Denote this input image by

\[ F_{\text{RGB}}^{(t)}[m, n] = O_{\text{RGB}}^{(t)}[m + i^{(t)}, n + j^{(t)}] \text{ for all } [m, n] \in \text{BR}_f^{(t)}, \tag{4.5} \]

where \([i^{(t)}, j^{(t)}]\) are the pixel coordinates for the origin of \(\text{BR}_f^{(t)}\) relative to the origin of \(O_{\text{RGB}}^{(t)}\). Note that \(F_{\text{RGB}}^{(t)}\) has the same dimensions as \(\text{BR}_f^{(t)}\).

**Colour feature map**

Figure 4.13 summarizes the first half of the mouth localization algorithm dedicated to the construction of a colour feature map.

![Colour Feature Extraction Diagram](Image)

**Figure 4.13:** First half of the mouth localization module.

The colour feature map is a fusion of two colour features \(F_1\) and \(F_2\). The colour features \(F_1\) and \(F_2\) are stored in 32-bit floating point matrices \(J_{F_1}^{(t)}\) and \(J_{F_2}^{(t)}\) respectively, while the colour feature map has representation as an 8-bit gray-scale image.

\(F_1\) is a RGB colour feature that incorporates red-green difference and green-blue difference. Specifically,

\[ J_{F_1}^{(t)}[m, n] = \begin{cases} J_{F_{R_1} - F_{G_1}}^{(t)}[m, n] - |J_{F_{G_1} - F_{B_1}}^{(t)}[m, n]|, & \text{if } [m, n] \in A \\ 0, & \text{otherwise}, \end{cases} \tag{4.6} \]

where

\[ J_{F_{R_1} - F_{G_1}}^{(t)}[m, n] = F_{R_1}^{(t)}[m, n] - F_{G_1}^{(t)}[m, n], \tag{4.7} \]

\[ J_{F_{G_1} - F_{B_1}}^{(t)}[m, n] = F_{G_1}^{(t)}[m, n] - F_{B_1}^{(t)}[m, n]. \tag{4.8} \]
and \( A = \{ [m,n] | F_R^{(t)}[m,n] \geq F_G^{(t)}[m,n] \text{ and } F_R^{(t)}[m,n] \geq F_B^{(t)}[m,n] \} \). The set \( A \) suppresses pixels that do not have red as the dominant colour component; such pixels are unlikely saturated red pixels. Because skin tone is mainly regulated by the red and green components [28], the red-green difference will separate the saturated reds of the lips and tongue from the skin. On the other hand, the skin typically expresses larger green-blue difference than the lips and tongue due to the larger green component contribution of skin tone. The green-blue difference at the lips and tongue tends towards zero because saturated red implies similarly minimal contributions from the green and blue components.

Note that \( F_1 \) attenuates achromatic pixels because \( F_R^{(t)}[m,n] = F_G^{(t)}[m,n] = F_B^{(t)}[m,n] \) implies \( J_{F_1}^{(t)}[m,n] = 0 \). To fit with the 8-bit representation of the colour feature map, \( J_{F_1}^{(t)} \) is further transformed by

\[
J_{F_1}^{(t)}[m,n] \leftarrow \min\{4 \times J_{F_1}^{(t)}[m,n], 255\},
\]

where \( \leftarrow \) is the (right-to-left) assignment operator from computer programming. Even though \( J_{F_1}^{(t)}[m,n] \in \{0,1,\ldots,255\} \) before the transformation, the values of \( J_{F_1}^{(t)} \) typically do not exercise the entire 8-bit range without amplification. A scaling factor of 4 is chosen so that \( J_{F_1}^{(t)} \) saturates towards the maximum value of 255 at the lip and tongue pixels. Values of \( J_{F_1}^{(t)} \) that exceed 255 after amplification are truncated to 255.

\( F_2 \) is also a RGB colour feature. But first, the input \( F_{rgb}^{(t)} \) is preprocessed on the HSV colour space to form \( F_{rgb}^{(t)} \), by

\[
W_{HSV}^{(t)} = \text{RGB2HSV}(F_{rgb}^{(t)})
\]

\[
W_S^{(t)}[m,n] \leftarrow \begin{cases} 
255, & \text{if } W_S^{(t)}[m,n] \geq 32 \text{ and } W_V^{(t)}[m,n] \geq 64 \\
W_S^{(t)}[m,n], & \text{otherwise}
\end{cases}
\]

\[
F_{rgb}^{(t)} = \text{HSV2RGB}(W_{HSV}^{(t)}).
\]

This HSV preprocessing step improves colour saturation in regions affected by mild to moderate amounts of specular reflection from lighting (e.g. saliva-coated lips and tongue). The condition in equation (4.11) forbids preprocessing of pixels that are too dark (low
value) and/or nearly achromatic (low saturation); the chromatic information at these pixels is noisy (see figure 4.9) and forcing full saturation can introduce false colours. Note that HSV preprocessing is not used for colour feature F1; the artificial saturation tends to reduce the separation of lips and tongue colours from skin tone on the dimension of red-green difference. Figure 4.14 shows an example of the HSV preprocessing step.

Figure 4.14: Example of HSV preprocessing on the face region of interest of a sample input image. Notice the lips, but also parts of the forehead and cheeks, become more saturated in red colour.

The colour feature F2 is computed as a ratio of the RGB components of $F_{d(t)}^{RGB}$. Specifically,

$$J_{F2}^{(t)}[m,n] = \frac{255}{4} \times \min \left\{ \frac{F_R^{d(t)}[m,n]}{F_G^{d(t)}[m,n] + F_B^{d(t)}[m,n] + 1}, 4 \right\}$$

(4.13)

This feature accentuates pixels with saturated red colour because the red component dominates both the green and blue components. Moving away from red, contributions from green and blue components increase and so the corresponding $J_{F2}^{(t)}$ value decreases. The offset of 1 in the denominator ensures that $J_{F2}^{(t)}[m,n]$ will never be undefined. Empirical observations revealed that the R:(G+B) ratio typically hovers around 1.5 for most pixels of $F_{d(t)}^{RGB}$ and rarely exceeds 4. Thus, R:(G+B) values larger than 4 are truncated.
to 4 so that a fixed scaling factor of $\frac{255}{4}$ can be applied to map $J_{F2}^{(t)}$ into the entire 8-bit range.

It is expected that the R:(G+B) ratio will be larger at the lips and tongue than at the skin. Therefore, the colour feature F2 can be boosted by removing the sample mean $J_{F2}^{(t)}$ value estimated from skin pixels and then re-amplifying the result, i.e.

$$J_{F2}^{(t)}[m, n] \leftarrow 4 \times (J_{F2}^{(t)}[m, n] - \hat{\mu}),$$

(4.14)

where the skin-segmented image $S^{(t)}_{\text{RGB}}$ from face localization is used to compute

$$\hat{\mu} = \frac{\sum_{p \in P} J_{F2}^{(t)}(p)}{|P|}$$

(4.15)

with $P = \{ [m, n] \in \text{BR}^{(t)}_f \mid S^{(t)}_{\text{RGB}}[m+i^{(t)}, n+j^{(t)}] \neq \tilde{0} \}$ and $[i^{(t)}, j^{(t)}]$ are the pixel coordinates for the origin of $\text{BR}^{(t)}_f$ relative to the origin of $O^{(t)}_{\text{RGB}}$. The notation $| \cdot |$ denotes cardinality of a set. The scaling factor of 4 is chosen so that the revised $J_{F2}^{(t)}$ values exercise the entire 8-bit range. Furthermore, $J_{F2}^{(t)}$ values are truncated at a minimum value of zero and a maximum value of 255. The re-amplification boosts lips and tongue colour accentuation by colour feature F2 because subtraction by $\hat{\mu}$ drops the $J_{F2}^{(t)}$ values at skin pixels closer to zero and so the re-amplification affects the skin pixels less.

The colour features F1 and F2 are unreliable in regions of $F^{(t)}_{\text{RGB}}$ that are darkened due to inherent face features (e.g. pupils, nostrils, oral cavity, hair, etc.) or shadows (e.g. under the nose, along the hairline, etc.). They are also unreliable in regions that are too bright and hence achromatic. Thus, the mouth localization algorithm calculates an 8-bit mask image $Y^{(t)}$ to inhibit responses of the colour feature map at these problematic regions. The mask image is computed as follows:

1. Construct the 8-bit gray-scale images

$$W_N^{(t)}[m, n] = \max\{F_R^{(t)}[m, n], F_G^{(t)}[m, n], F_B^{(t)}[m, n]\}$$

(4.16)

and

$$W_D^{(t)}[m, n] = W_N^{(t)}[m, n] - \min\{F_R^{(t)}[m, n], F_G^{(t)}[m, n], F_B^{(t)}[m, n]\}.$$  

(4.17)
$W_M^{(t)}$ provides a measure of brightness while $W_D^{(t)}$ provides a measure of achromaticity ($W_D^{(t)}[m, n] = 0$ implies an achromatic pixel at $[m, n]$).

2. Construct a gray-scale histogram $H^{(t)}(h \in \{0, 1, \ldots, 255\})$ with 256 bins from $W_M^{(t)}[m, n]$, using only pixels that correspond to skin (i.e. $[m, n]$ satisfies $S_{rgb}^{(t)}[m + \hat{i}^{(t)}, n + \hat{j}^{(t)}] \neq 0$ and $[i^{(t)}, j^{(t)}]$ are the pixel coordinates for the origin of $BR_j^{(t)}$ relative to the origin of $O_{rgb}$).

3. Find the 2.5th percentile $h_{2.5}^{(t)}$ and the 25th percentile $h_{25}^{(t)}$ of $H^{(t)}$.

4. Compute the threshold $d_M^{(t)} = h_{2.5}^{(t)} + 0.25 \times (h_{25}^{(t)} - h_{2.5}^{(t)})$.

5. Construct the 8-bit mask image

\[
Y^{(t)}[m, n] = \begin{cases} 
255, & \text{if } W_M^{(t)}[m, n] > d_M^{(t)} \text{ and } W_D^{(t)}[m, n] > d_D \\
0, & \text{otherwise}.
\end{cases}
\]  

The fixed threshold $d_D$ is set at 16. Using a fixed threshold to filter out achromatic pixels is appropriate because the tolerance for achromaticity typically does not change with camera configuration or the environment. On the other hand, a dynamic threshold $d_M^{(t)}$ is more appropriate to filter out dark and shadowed regions. First, it is to compensate for biasing from the camera brightness setting. Second, the definition of shadow is relative to illumination of the rest of the face.

Averaging the colour features $F_1$ and $F_2$ forms the colour feature map $X^{(t)}$, i.e.

\[
X^{(t)}[m, n] = \begin{cases} 
\frac{1}{2} \times (J_{F1}^{(t)}[m, n] + J_{F2}^{(t)}[m, n]), & \text{if } Y^{(t)}[m, n] \neq 0 \\
0, & \text{otherwise}.
\end{cases}
\]  

Figure 4.15 shows an example of the colour feature map, including the colour features $F_1$ and $F_2$ visualized as images.

**Thresholding and localization**

Figure 4.16 summarizes the second half of the mouth localization module.
Figure 4.15: The colour feature map inside the face region of interest. This set of images exemplifies the action of the colour feature map on a closed mouth sample image.

The lips and tongue, if they are visible in $F^{(t)}_{\text{RGB}}$, should be highlighted by pixels of $X^{(t)}$ with high gray-scale intensity. Most of the other $X^{(t)}$ pixels should have lower intensity, because either they associate with skin colour or they are masked out. Therefore, $X^{(t)}$ can be processed further via gray-scale thresholding to form the refined colour feature map $X^{(t)}$, i.e.

$$X^{(t)}[m,n] = \begin{cases} X^{(t)}[m,n], & \text{if } X^{(t)}[m,n] > d_x^{(t)} \\ 0, & \text{otherwise.} \end{cases}$$

The refinement process discards lower intensity $X^{(t)}$ pixels that unlikely associate with
the lips and tongue. The dynamic threshold $d_k^{(t)}$ is computed as follows:

1. Construct a gray-scale histogram $H^{(t)}(h \in \{0, 1, \ldots, 255\})$ with 256 bins from $X^{(t)}[m, n]$, using only pixels that are kept for further analysis (i.e. $[m, n]$ satisfies $Y^{(t)}[m, n] \neq 0$). Normalize the histogram by the number of samples so that $\sum_h H^{(t)}(h) = 1$.

2. Apply Otsu’s thresholding [31] on $H^{(t)}$ to obtain the candidate threshold $b_{\text{Otsu}}$. For gray-scale images that have bi-modal gray-scale histograms (i.e. two classes or clusters), Otsu’s method finds the optimal threshold that minimizes the total within-class gray-scale variances of the two classes. Empirical observations revealed that the higher intensity and lower intensity pixels of $X^{(t)}$ usually separate into two clusters.

3. Temporal averaging: $d_{\text{Otsu}}^{(t)} = 0.8 \times d_{\text{Otsu}}^{(t-1)} + 0.2 \times b_{\text{Otsu}}$, to stabilize the Otsu threshold from sudden and inconsequential changes.

4. Compute a second candidate threshold

$$b_{\text{Tail}} = \arg \max_{c \in C} \{H^{(t)}(c + 1) - H^{(t)}(c)\}$$

(4.21)

where $C = \{h \in \{1, 2, \ldots, 254\} \mid H^{(t)}(h) > 0.005\}$. Because there are many more
skin pixels (especially when the lips and tongue are not visible), sometimes the high intensity cluster may be ill-formed or even non-existent. In these cases, \( H(t) \) will no longer be bi-modal and so \( d_{\text{otsu}}(t) \) may not be appropriate. The purpose of \( b_{\text{tail1}} \) is to estimate the tail of the low intensity cluster in \( H(t) \) so that most of the low intensity pixels can still be thresholded away.

5. Temporal averaging:
\[
d_{\text{tail}}(t) = 0.8 \times d_{\text{tail}}(t-1) + 0.2 \times b_{\text{tail1}},
\]
to stabilize the tail threshold from sudden and inconsequential changes.

6. Compute
\[
d_{\text{X}}(t) = \frac{1}{2} \times (d_{\text{otsu}}(t) + d_{\text{tail}}(t)).
\] (4.22)

Let \( \text{BR}^{(t)}_{m} \) represent the bounding rectangle of the mouth region of interest, defined relative to \( \text{I}_{\text{RGB}}^{(t)} \). Note that \( \text{BR}^{(t)}_{m} \subset \text{BR}^{(t)}_{f} \). This bounding rectangle is constructed by connected component analysis [41] on the refined colour feature map \( X(t) \) as follows:

1. Create \( X^{n(t)} \), an exact copy of \( X(t) \).

2. Find \( [i,j] = \arg \max_{[m,n]} \{X^{n(t)}[m,n]\} \), the most intense \( X^{n(t)} \) pixel.

3. If \( X^{n(t)}[i,j] < 128 \), terminate the analysis.

4. Otherwise, construct a candidate connected component \( Q \) by 8-way connected component analysis starting with the pixel \([i,j] \) [41]. For each new pixel added to \( Q \), examine its 8 adjacent pixels and add to \( Q \) any of these pixels with gray-scale intensity in \([X^{n(t)}[i,j] - 30, X^{n(t)}[i,j] + 30] \). Thus, \( Q \) consists of all pixels neighbouring \([i,j] \) that have similar intensity as \( X^{n(t)}[i,j] \).

5. For each pixel in \( Q \), set the corresponding pixel in \( X^{n(t)} \) to zero. This removes from \( X^{n(t)} \) the pixels already belong to a candidate connected component.

6. Compute the smallest bounding rectangle \( \text{BR}_{cc} \) that still contains \( Q \) entirely.
7. Discard $\text{BR}_{cc}$ if its width and height are both less than 10 pixels or the cardinality of $Q$ is less than 50. Such bounding rectangle and connected component are too small to contain significant face features such as lips or tongue.

8. Otherwise, add $\text{BR}_{cc}$ to the list of candidate bounding rectangles.

9. Repeat steps 2–8 for 25 times, producing the set of candidate bounding rectangles $\{\text{BR}_{cc}^{(1)}, \ldots, \text{BR}_{cc}^{(k)}\}$ for $0 \leq k \leq 25$.

$\text{BR}_{m}^{(t)}$ begins as the candidate bounding rectangle $\text{BR}_{cc}^{(p)}$ whose centroid is closest in Euclidean distance to the centroid of the previous mouth region of interest bounding rectangle $\text{BR}_{m}^{(t-1)}$. This follows from the observation that the new mouth position ought to be nearby the previous position, as long as the video frame rate is not too slow. However, $\text{BR}_{cc}^{(p)}$ alone may not contain the entire mouth. For example, when the mouth is opened and the two lips are separated in $X(t)$, the connected component $Q^{(p)}$ may capture only one lip and so $\text{BR}_{cc}^{(p)}$ will not be covering the entire mouth. Therefore, $\text{BR}_{m}^{(t)}$ aggregates other candidate bounding rectangles whose centroids are less than Euclidean distance of $d_{cc}$ units away from the centroid of $\text{BR}_{m}^{(t-1)}$. The threshold $d_{cc} = \max\{0.65 \times \sqrt{\text{Width}(\text{BR}_{m}^{(t-1)})^2 + \text{Height}(\text{BR}_{m}^{(t-1)})^2}, 25\}$.

The mouth is deemed not found in $F_{\text{RGB}}^{(t)}$ if both the width and height of $\text{BR}_{m}^{(t)}$ are less than 20 pixels. Such $\text{BR}_{m}^{(t)}$ is too small to be associated with the mouth. In this case, the switch output will be inhibited (see section 4.3.3).

Figure 4.17 shows some results of the mouth localization module for three example scenarios.

4.3.3 Tongue switch

Recall the tongue switch module’s functions are 1) to calculate a tongue-modulated statistic $z_{\text{TMS}}^{(t)}$ for switch activation and 2) to produce a frontal view measure $z_{\text{PVM}}^{(t)}$ that corresponds to how well the camera is perceiving a frontal view of the user’s face.
Tongue-modulated statistic

It is reasonable to assume that the user’s head will be relatively calm whenever he or she is ready for switch activation. The user may not necessarily be facing the camera due to spasticity, but at least the head pose will be stable. If visible spastic head movements are present, the user will typically be too preoccupied to concentrate on switch activation.

The observation above implies that the amount of saturated red pixels in the mouth region of interest tends to be stable just prior to switch activation. Contributions to this baseline count primarily come from pixels of the lips. Protruding the tongue increases the amount of saturated red pixels in the mouth region of interest beyond typical baseline fluctuations. Measuring the changes to the amount of saturated red pixels in the mouth region of interest is the basis of the tongue-modulated statistic.

Conveniently, the colour feature map $X^{(t)}$ from mouth localization captures all the saturated red pixels in the face region of interest, which includes the mouth region of interest. Therefore, changes to the amount of saturated red pixels in the mouth region of interest can be measured as follows:
1. Compute $n_{SRP}^{(t)}$, a count of non-zero pixels in $X^{(t)}$ for all pixels contained in the mouth region of interest (i.e. for all pixels $[m, n] \in BR_m^{(t)}$).

2. Compute the sample average $\bar{n}_{SRP}$ from the past $i$ counts, i.e.

$$\bar{n}_{SRP} = \frac{\sum_{j=t-i}^{t-1} n_{SRP}^{(j)}}{i}.$$  \hspace{1cm} (4.23)

This average represents an estimate of the baseline count. The analysis window size $i$ is set to 15 frames, which approximates to 1 second of analysis due to a frame rate of 15 Hz. Setting the analysis window too long slows the estimate’s response to actual baseline changes, while setting it too short makes the estimate unreliable because of insufficient samples.

3. The measured change is calculated by $n_{SRP}^{(t)} - \bar{n}_{SRP}$.

Note that opening of the mouth can also induce a similar increase of red pixels in the mouth region of interest because it unveils the retracted tongue. But opening of the mouth exposes the oral cavity as well, while a true tongue protrusion hardly reveals the oral cavity. Recall that oral cavity pixels are marked in the mask image $Y^{(t)}$ from mouth localization. Therefore, to resolve the confusion between tongue protrusion and mouth opening, the increase of oral cavity pixels due to mouth opening can be used to cancel the increase of red pixels. The tongue-modulated statistic $z_{TMS}^{(t)}$ is completed as follows:

1. Calculate $n_{SRP}^{(t)}$ and $\bar{n}_{SRP}$ as before.

2. Compute $n_{OCP}^{(t)}$, a count of non-zero pixels in $Y^{(t)}$ for all pixels contained in the mouth region of interest. The value of $n_{OCP}^{(t)}$ should be large for an opened mouth and much smaller during a tongue protrusion.

3. $z_{TMS}^{(t)} = \max\{ (n_{SRP}^{(t)} - \bar{n}_{SRP}) - \frac{n_{OCP}^{(t)}}{5}, 0 \}$. 
The attenuation factor of $\frac{1}{5}$ on $n_{OCP}^{(t)}$ is necessary because mouth corners may be present even during tongue protrusion. The mouth corner pixels contribute to $n_{OCP}^{(t)}$. Without the attenuation factor, $n_{OCP}^{(t)}$ can dominate $n_{SRP}^{(t)} - \bar{n}_{SRP}$.

The statistic $z_{TMS}^{(t)}$ is sensitive to the protrusion motion of the tongue. If the tongue remains protruded, $z_{TMS}^{(t)}$ converges back to zero because there is little change in the amount of red pixels; the tongue pixels are temporarily incorporated into the baseline count. During tongue retraction, the number of red pixels in the mouth region of interest decreases, making $n_{SRP}^{(t)} - \bar{n}_{SRP}$ a negative value and so $z_{TMS}^{(t)}$ goes to zero.

Frontal view measure

Most of the time, the user’s head will be closer to the upright orientation instead of being rolled to the sides. In terms of face and mouth bounding rectangles $BR_f^{(t)}$ and $BR_m^{(t)}$ respectively, $BR_m^{(t)}$ should be near the left or right boundaries of $BR_f^{(t)}$ when the head is turned to the sides. Moreover, when the frontal view of the face is shown, $BR_m^{(t)}$ should be close to the line that bisects $BR_f^{(t)}$ into left and right halves.

The frontal view measure $z_{FVM}^{(t)}$ follows exactly the aforementioned observations. Suppose the pixel coordinates $[i^{(t)}, j^{(t)}]$ mark the centroid of $BR_m^{(t)}$ relative to the origin of $BR_f^{(t)}$. Then,

$$z_{FVM}^{(t)} = 0.7 \times z_{FVM}^{(t-1)} + 0.3 \times b_{FVM},$$

where

$$b_{FVM} = 1 - \frac{|i^{(t)} - \frac{1}{2} \times \text{Width}(BR_f^{(t)})|}{\frac{1}{2} \times \text{Width}(BR_f^{(t)})}.$$  \hspace{1cm} (4.25)

Note that $z_{FVM}^{(t)} \in [0, 1]$. Furthermore, $z_{FVM}^{(t)} \rightarrow 0$ as the centroid of $BR_m^{(t)}$ moves away from the bisector line and goes closer to the left and right boundaries of $BR_f^{(t)}$. Larger $z_{FVM}^{(t)}$ implies the camera is better perceiving a frontal view of the user’s face.
Switch activation and inhibition

If the single camera algorithm is used alone, switch activation is achieved by positive edge thresholding of the tongue-modulated statistic. This means a switch activation is issued if and only if $z_{TM}^{(t)}$ rises past the switch activation threshold $d_{activation}$, i.e.

$$z_{TM}^{(t-1)} < d_{activation} \text{ and } z_{TM}^{(t)} > d_{activation} \iff \text{switch activation on frame } t \quad (4.26)$$

The choice of $d_{activation}$ is crucial with regards to switch sensitivity and specificity. Setting it too small will force many false activations. Setting it too large can render the algorithm unresponsive to all tongue protrusions.

Putting the switch into inhibition simply means setting $z_{TM}^{(t)} = 0$, setting $z_{FVM}^{(t)} = 0$, and setting a switch inhibition status indicator $z_{inhibit}^{(t)} \in \{0, 1\}$ to 1. Upon recovery, it is best to delay resetting $z_{inhibit}^{(t)}$ back to 0 until after a few frames, thus giving $z_{TM}^{(t)}$ and $z_{FVM}^{(t)}$ a chance to stabilize.

4.3.4 Motion-based inhibitor

For technical and clinical reasons, the tongue switch ought to be inhibited when the user experiences head and neck spasticity. The single camera algorithm can temporarily lose track of the mouth due to rapid motion from spastic head movements. Fluctuations to the tongue-modulated statistic $z_{TM}^{(t)}$ associate with gross motor movements rather than tongue protrusions. Also, during spasticity the user will likely be too distracted by discomfort to concentrate on switch activation.

When the user experiences head and neck spasticity, the spastic head movements cause motion in the face region of interest. The motion-based inhibitor disables the tongue switch when it measures too much motion in the face region of interest. Indeed, the motion cues are not specific to head movements. This is acceptable because other sources of motion may also be distractors affecting reliable operation. For example, arm movements across the face region of interest may temporarily occlude the mouth.
The inhibitor maintains a motion history image [11] from the input frames. Motion history image is a gray-scale image that maps regions of the frame where motion was recently detected. Pixels with highest gray-scale intensity correspond to positions where motion has just occurred in the past frame. If the motion is not sustained, these pixel intensities decay to zero at a rate inversely proportional to the duration ($u_{\text{duration}}$) in seconds that the motion history is to be maintained. Thus, the motion history image records traces of change that occurred in the most recent $u_{\text{duration}}$ seconds (see figures 4.18 and 4.19 for two examples).

Figure 4.18: Example of motion history image capturing left-to-right head displacement. The more intense the green pixels, the more recent the motion was at those pixels.

Figure 4.19: Example of motion history image capturing right-to-left head rotation. The more intense the green pixels, the more recent the motion was at those pixels.

The motion-based inhibitor is summarized by the following steps:
1. Convert original input frames $O^{(t)}_{\text{RGB}}$ and $O^{(t-i)}_{\text{RGB}}$ (note: $0 < i < t$) to 8-bit gray-scale images. The prototype uses $i = 2$.

2. Compute the silhouette image

$$W^{(t)}_{\text{silhouette}}[m,n] = \begin{cases} 
1, & \text{if } |O^{(t)}_{\text{gray}} - O^{(t-i)}_{\text{gray}}| > d_{\text{motion}} \\
0, & \text{otherwise}
\end{cases}$$ (4.27)

Thus, a pixel is said to have “motion” if the change in gray-scale intensities between the two input frames at that pixel exceeds the motion threshold $d_{\text{motion}}$. The silhouette image marks the pixels where motion is detected. A $d_{\text{motion}}$ value of 30 is sufficient to detect spastic head movements. It is large enough to ignore the baseline fluctuations of the gray-scale intensities due to the electronics of the imaging elements.

3. Time-stamp the current time $u$.

4. Update a time-stamp matrix

$$J^{(t)}[m,n] = \begin{cases} 
u, & \text{if } W^{(t)}_{\text{silhouette}}[m,n] \neq 0 \\
0, & \text{if } W^{(t)}_{\text{silhouette}}[m,n] = 0 \text{ and } J^{(t-1)}[m,n] < u - \nu_{\text{duration}} \\
J^{(t-1)}[m,n], & \text{otherwise}
\end{cases}$$ (4.28)

In particular, the first assignment time-stamps the pixels where motion has just been detected, while the second assignment invalidates pixels where motion has not been detected for more than $\nu_{\text{duration}}$ seconds. A $\nu_{\text{duration}}$ of 1 second matches well with the short durations of typical spastic head movements.

5. Compute the motion history image

$$W^{(t)}_{\text{MHI}}[m,n] = 255 \times \max\left\{\frac{\nu_{\text{duration}} - (u - J^{(t)}[m,n])}{\nu_{\text{duration}}}, 0\right\}.$$ (4.29)

The scaling factor of 255 corresponds to maximum intensity of an 8-bit gray-scale image.
6. Apply the bounding rectangle $\text{BR}^{(t)}_f$ on $W^{(t)}_{\text{MHI}}$ to focus on the face region of interest. Compute the motion inhibit measure

$$m^{(t)}_{\text{inhibit}} = \frac{|\{(m, n) \in \text{BR}^{(t)}_f \mid W^{(t)}_{\text{MHI}}[m, n] \neq 0\}|}{\text{Area}(\text{BR}^{(t)}_f)},$$  \hspace{1cm} (4.30)

where $|\cdot|$ denotes cardinality of a set. In other words, $m^{(t)}_{\text{inhibit}} \in [0, 1]$ measures the fraction of pixels inside the face region of interest that have undergone motion.

7. Inhibit the switch output (see section 4.3.3) if $m^{(t)}_{\text{inhibit}} > d_{\text{inhibit}}$. Setting $d_{\text{inhibit}}$ too small renders the inhibitor sensitive to the slightest of movement. Setting $d_{\text{inhibit}}$ too large makes the inhibitor ineffective. The prototype uses a $d_{\text{inhibit}}$ value of 0.65.

### 4.4 Fusion algorithm

The fusion algorithm of the multiple camera tongue switch follows the best camera paradigm. At any given time, the fusion algorithm decides on switch activation using data coming from the camera that has the best frontal view of the user’s face.

Let $z^{(t)}_{\text{MCS}}$ represent the multiple camera statistic that decides switch activation for the multiple camera tongue switch. The fusion algorithm inputs are the sets of tongue-modulated statistics $\{z^{(t)}_{\text{TMS}_1}, z^{(t)}_{\text{TMS}_2}, \ldots, z^{(t)}_{\text{TMS}_p}\}$, frontal view measures $\{z^{(t)}_{\text{FVM}_1}, z^{(t)}_{\text{FVM}_2}, \ldots, z^{(t)}_{\text{FVM}_p}\}$, and inhibition indicators $\{z^{(t)}_{\text{inhibit}_1}, z^{(t)}_{\text{inhibit}_2}, \ldots, z^{(t)}_{\text{inhibit}_p}\}$, for $p$ cameras. The steps of the fusion algorithm are as follows:

1. Construct the index set $C^{(t)} = \{c \in \{1, \ldots, p\} \mid z^{(t)}_{\text{FVM}_c} > d_{\text{FVM}}$ and $z^{(t)}_{\text{inhibit}_c} = 0\}$. This set discards cameras that have unsatisfactory views of the face or are inhibited.

The frontal view threshold $d_{\text{FVM}}$ is set at 0.5. This corresponds to the head being rotated to the left or right by $30^\circ$, as shown in figure 4.20.
Figure 4.20: An overhead view of the user’s head. The numerator in equation (4.25) of the $z_{FVM}^{(t)}$ calculations can be seen as a projection of the mouth position along the width dimension of the frame (or imaging) plane. The derivations above show that the $b_{FVM}$ value of 0.5 corresponds to lateral head rotation by $30^\circ$ away from the user-camera axis (i.e. the normal of the frame plane).
2. Compute

\[ z_{\text{MCS}}^{(t)} = \begin{cases} 
0, & \text{if } C^{(t)} = \emptyset \\
\hat{z}_{\text{TMS}}^{(t)}, & \text{where } k = \arg \max_{c \in C^{(t)}} \{ \hat{z}_{c}^{(t)} \}. 
\end{cases} \] (4.31)

3. Switch activation is achieved by positive edge thresholding of \( z_{\text{MCS}}^{(t)} \). Namely,

\[ z_{\text{MCS}}^{(t-1)} < d_{\text{activation}} \text{ and } z_{\text{MCS}}^{(t)} > d_{\text{activation}} \iff \text{switch activation on frame } t \] (4.32)

The parameter \( d_{\text{activation}} \) should be exposed for tuning during field deployment, in order to adjust for switch sensitivity and specificity (see section 4.3.3). Also, appropriate values of \( d_{\text{activation}} \) depend on scaling of the user’s face inside the cameras’ fields of view. The prototype uses a \( d_{\text{activation}} \) value of 250.

### 4.5 Example software implementation

The multiple camera tongue switch prototype is primarily a software implementation; the single camera algorithm, the fusion algorithm, and controls to the tongue switch are all realized by software implementations running on a general purpose processor instead of hardware implementations on application specific hardware. Many possible software implementations exist due to the major operating systems (i.e. Microsoft Windows, Mac OS X, and GNU/Linux) supporting different image processing libraries and video application frameworks. The following subsections summarize the Microsoft Windows-based software implementation featured in the prototype.

#### 4.5.1 Model-view-controller approach

The prototype’s software implementation follows the model-view-controller (MVC) architecture [27]. The model contains the software components that manipulate application data. The view handles presentation of application data and provides an interface for user actions. The controller bridges the model and the view by processing user actions and
responding to software events from the model. Typically, the controller is separate from
the view because one controller may be supporting multiple views implementing multiple
presentation modalities (e.g. a graphical interface and a command-line interface).

Figure 4.21 shows the prototype software’s MVC structure and the relationships of
the software components with the switch user, the caregiver, and the switch hardware.

Figure 4.21: The prototype’s software implementation using the model-view-controller
architecture. The dash line separates the model domain from the view/controller domain.

In particular, the multiple camera tongue switch resides completely in the model
domain. The MVC workflow exists only to support an operator interface for switch
control. Because the tongue switch has only one user interface, the view and controller
are merged into a single entity. Furthermore, the operator interface is intended for the
individual that sets up the system for the switch user. This individual, fulfilling the role
of switch operator, is usually a caregiver of the switch user.
4.5.2 View: the operator interface

To deploy the multiple camera tongue switch, the switch operator needs to control the run-time behaviour of the switch, configure the camera optics, and visualize the video stream of each camera for field-of-view aiming. The prototype’s operator interface, shown in figure 4.22, supports the aforementioned operator tasks.

Figure 4.22: The switch operator’s user interface. From the left, there are the three channel control panels. The linked controls for camera configuration are located at the bottom right corner. The tongue switch controls occupy the top right corner.

This interface features three identical channel control panels. Each channel panel contains controls to enable the channel and to select the channel camera, a viewing area for real-time visualization of the camera’s video feed, and controls to configure the camera’s optics, which include settings for zoom, exposure, brightness, and white balance. By disabling the appropriate number of channels, the prototype can be reconfigured into a one-, two-, or three-camera system. Linked controls from a fourth panel allow two or more cameras to share the same zoom, brightness, and white balance settings. These controls are particularly useful for conditioning the video streams identically, when identical types of cameras are used. Finally, there is a control to start and stop the tongue switch and
another to refresh the list of available cameras.

The operator interface above is part of a Microsoft Foundation Classes (MFC) application [20]. The MFC application framework facilitates development of applications that follow the MVC architecture. In particular, MFC provides most of the controller layer logic to drive the operator interface shown in figure 4.22.

4.5.3 Model: the tongue switch domain

The model should process in parallel to the operator interface in a separate thread of execution. It should be continuously processing frames from the input video streams regardless of the state of the operator interface (except when the switch software shuts down). Microsoft’s DirectShow framework for multimedia applications facilitates such an implementation of the model. The framework manages multimedia data via filter graphs. Associated with a filter graph are one or more video and/or audio streams and a collection of filters that typically multiplex, demultiplex, or transform data streams. DirectShow follows the pipe-and-filter architecture [27] with regards to stream processing. Each stream passes through one or more filters before arriving at its final output, typically a renderer or a file writer. The filter graph component automatically handles the low-level details of building and managing these stream processing graphs, such as allocating stream buffers between filters, synchronizing multiple streams during multiplexing and demultiplexing, and scheduling processor time among the filters. A filter graph has its own thread of execution; starting and stopping a filter graph respectively resumes and suspends all of the filter graph’s filters.

Figure 4.23 demonstrates the multiple camera tongue switch implementation using a filter graph.

The figure refers to a three-camera system. The cameras relate to separate instances of an identically structured stream processing graph. The frame grabber filter prepares an inbound RGB video stream of $320 \times 240$ pixels/frame at 15 frame/second. Next, the
Figure 4.23: Details of the model domain. The camera processor software component hosts the algorithms of the multiple camera tongue switch. The shaded boxes identify DirectShow filters.

Video stream passes through a custom DirectShow transform filter containing the single camera algorithm. Finally, the video stream is rendered at the operator interface, in the viewing area of the appropriate channel. The custom DirectShow filter has access to a data buffer outside of the filter graph. At the end of each frame processing, the custom filter posts the latest $z_{TNS}^{(t)}$, $z_{FWM}^{(t)}$, and $z_{\text{inhibit}}^{(t)}$ (see section 4.3.3) onto this buffer. The filter graph ensures fair share of processor time for all three stream processing graphs. Note that this implementation easily supports $N$ cameras: just instantiate $N$ stream processing graphs and data buffers.

The data buffers are also connected to a passive software component that implements the fusion algorithm. When invoked, the component reads the latest single camera scores from the data buffers and then calculates $z_{MCS}^{(t)}$ (see section 4.4).

A switch component and a trigger component (figure 4.24) complete the software implementation. The passive switch component provides a façade for switch output.
When invoked, this component generates, depending on configuration, a software output or a hardware output via toggling a serial port pin. The active trigger component bridges the fusion algorithm to the switch output and it has its own thread of execution. Until the switch software shuts down, the trigger perpetually executes a simple loop; it first invokes the fusion algorithm for $z_{MCS}^{(t)}$, and then it applies positive edge thresholding on $z_{MCS}^{(t)}$ for switch activation (see section 4.4).

Figure 4.24: Further details of the model domain. The trigger software component implements a perpetual loop. The numbered items cycle through the loop actions.

### 4.5.4 Image processing and performance enhancement libraries

The single camera algorithm implementation inside the custom DirectShow filter makes use of the OpenCV v1.0 library [4]. OpenCV provides implementations of numerous data structures and algorithms in image processing and computer vision, including the CAMSHIFT algorithm.

To speed up image processing, the software implementation is compiled with Intel’s Integrated Performance Primitives (IPP) v5.1 package. IPP provides processor architecture and data type optimized implementations of popular signal and image processing algorithms. OpenCV automatically uses IPP optimizations if the package is available [4].
Chapter 5

Usability Experiment

The objectives of the usability experiment were:

1. to evaluate performance of the multiple camera tongue switch prototype, in terms of sensitivity and specificity towards intentional tongue protrusions in a controlled environment, and

2. to determine the utility of the peripheral cameras with regards to improving single-switch access.

To address these objectives, one study participant completed five experiment sessions where he worked on a single-switch activity using the tongue switch prototype.

Furthermore, this usability experiment was intentionally made to focus on the prototype (the product) rather than the participant (the user), in order to investigate whether the performance of the tongue switch prototype was adequate as an access solution for the participant. User-centric evaluations that typically accompany usability studies were beyond the scope of this experiment and thus user-centric usability measures were not included.
5.1 Protocol

5.1.1 Preconditions

The participant had past experience working with computers, completing multiple choice exercises on a classroom computer with caregiver assistance. Furthermore, the participant received three months of tongue protrusion training, averaging two 30-minute sessions per week, via a training regime modified from caregiver-mediated access (see section 3.1). As such, he was already proficient with voluntary tongue protrusions prior to the first experiment session.

5.1.2 Session format, scheduling, and location

The experiment sessions consisted of the participant working through multiple repetitions of a single-switch activity using the tongue switch prototype. The single-switch activity remained the same for all repetitions and for all sessions. A caregiver was present during each session to assist with implementing the experiment protocol. The same caregiver was brought in for all sessions. In particular, this caregiver was very familiar with the participant’s behaviours and responses.

The first step of each session was to position the participant in front of the multiple camera tongue switch according to the user and system setup outlined in section 4.2.1. After calibrating the cameras (see section 4.5.2), the single-switch activity commenced. The participant alternated between periods of single-switch activity and rest. Each activity period lasted no more than two minutes, while each rest period had a minimum duration of two minutes. The short activity periods were intentional in order to minimize the participant’s fatigue level and to maximize his attention. The longer rest periods allowed the participant to recover his strength, especially for tongue protrusions. Whenever the participant could not continue with the activity due to severe discomfort from spasticity, the activity period was terminated and the results for that period were
discarded. Each experiment session lasted no more than one hour. There was no quota on the number of successful repetitions per session. After all, the participant’s physical condition varied from one day to another due to cerebral palsy.

The participant completed five sessions over the course of the usability experiment. These five sessions were held on five different days. Each session was scheduled for at least one week after the previous. The time of day for a session was based on the participant’s and caregiver’s availability.

All five experiment sessions were conducted in the same room. This room had enough space to comfortably accommodate three people (the experimenter, the caregiver, and the participant in his wheelchair) and the experimental apparatus. With regards to lighting, overhead fluorescent tubes were the only light sources of this room. There were no overhead structures casting shadows on the participant and there were no windows for sunlight. The backdrop around the participant consisted of white walls and some furniture. All objects resembling skin colour were temporarily relocated elsewhere outside of the backdrop; this was to minimize interference with face localization (see section 4.3.1).

Finally, the participant was asked not to wear red-coloured clothing for any of the sessions, in order to minimize interference with mouth localization (see section 4.3.2).

5.1.3 Single-switch activity

The single-switch activity was one with which the participant was familiar. This activity took the form of picture matching. The participant had experience with word identification and matching from his classroom computer work; picture matching was chosen instead because it was easier and quicker for him to perceive graphics than to read words.

The single-switch picture matching activity was implemented as a custom program in MATLAB. The program had a collection of photographs, graphics, and clip art totaling 30 items. For each run, the program randomly chose 10 items from the collection to form the working set. From the working set, one item was selected as the target picture. Next,
the working set was extended to 13 items by duplicating the target picture three more times. The program then constructed a choice vector by choosing with replacement 20 items from the extended working set. Therefore, the purpose of the extended working set was to increase the probability of adding the target picture into the choice vector. Finally, the program presented the choice vector pictures one-by-one alongside the target picture each time. The participant was asked to make a switch activation if and only if the choice picture matched the target picture. Each choice was displayed for three seconds, shorter if the program detected a switch activation. The program was responsive to left mouse clicks.

Usually, the choice vector contained many more non-target pictures than instances of the target picture. This is reasonable because there are typically many more non-actionable rather than actionable cues in single-switch one-of-n selection tasks [30]. Furthermore, there was no a priori knowledge of the multiplicity of target picture instances, in the choice vector of each run. Coupled with the one-by-one presentation of elements in the choice vector, the program made it impossible for the participant to anticipate switch activations.

The program played an audio cue each time the program registered a left mouse click. This was to alert the participant of a successful switch activation. It was the only audio feedback that the program produced.

5.1.4 Apparatus

The key apparatus was the multiple camera tongue switch prototype (see section 4.2). The picture matching program ran on another desktop computer equipped with MATLAB. By hosting the prototype and the picture matching program on separate computers, there were no compromises to the performances of both applications.

Figure 5.1 shows a computer mouse that was modified with an electronic trigger for left mouse clicks. The modified mouse interfaced with the prototype’s single-switch
output (see section 4.2.4) through a 1/4-inch mono connection. Pulling the single-switch output to logical LOW was equivalent to depressing the left mouse button.

Figure 5.1: A modified computer mouse featuring a mono plug for an electronic trigger.

The tongue switch prototype interfaced with the picture matching program via the modified computer mouse. The modified mouse was connected to the desktop computer that hosted the picture matching program. In fact, this setup follows exactly the example hardware deployment shown in figure 4.3. Figures 5.2a and 5.2b show the setup from one of the experiment sessions.

Figure 5.2: A typical experiment session.
5.1.5 Data collection

During each activity period, the picture matching program collected the following data:

- The chosen target picture, denoted by its picture index.

- The choice vector, with picture indices as its elements.

- A switch activation history vector of the same length as the choice vector and with elements in \( \{0, 1\} \). The \( i \)th element of this vector was set to 1 if and only if the program registered a left mouse click during the presentation of the \( i \)th choice.

Independently, the tongue switch prototype wrote the following into a time-stamped prototype log file:

- A new entry for each and every time the trigger component invoked the fusion algorithm (see section 4.5.3). These entries recorded fusion algorithm inputs at the time of invocation, which included the tongue-modulated statistic \( z_{\text{TMS}}^{(t)} \), frontal view measure \( z_{\text{FVM}}^{(t)} \), and inhibition indicator \( z_{\text{inhibit}}^{(t)} \) from all three cameras.

- Entries that marked when the picture matching program presented a new choice. These entries were manually issued by the experimenter, via a button of the prototype’s operator interface, whenever he observed the new choices.

- Entries that marked when the participant protruded his tongue. These entries were manually issued by the experimenter via another button of the operator interface, but the experimenter was cued to do so by the caregiver. During the activity periods, the caregiver had the responsibility of observing the participant for intentional tongue protrusions. The caregiver was entrusted with this task because she was very familiar with the participant.
5.2 Data analysis and performance metrics

The tongue switch prototype’s performance was quantified in terms of sensitivity and specificity towards intentional tongue protrusions. Values for these two metrics were calculated from the counts of true positive instances \( n_{\text{TP}} \), false negative instances \( n_{\text{FN}} \), true negative instances \( n_{\text{TN}} \), and false positive instances \( n_{\text{FP}} \). In this experiment, the definitions for these four cases were as follows:

- **True positive**: a switch activation with tongue protrusion from the participant.
- **False negative**: switch inaction despite a tongue protrusion from the participant.
- **True negative**: switch inaction corresponding to the absence of tongue protrusion.
- **False positive**: a switch activation in the absence of tongue protrusion.

Sensitivity, with definition

\[
\text{sensitivity} = \frac{n_{\text{TP}}}{n_{\text{TP}} + n_{\text{FN}}} \times 100\%, \tag{5.1}
\]

measured the prototype’s responsiveness to tongue protrusions. Specificity, with definition

\[
\text{specificity} = \frac{n_{\text{TN}}}{n_{\text{TN}} + n_{\text{FP}}} \times 100\%, \tag{5.2}
\]

measured how often the switch activations corresponded to actual tongue protrusions. In general, a well performing switch would score high in both sensitivity and specificity.

There were two fundamental scenarios to the picture matching activity. In a *matching* scenario (figure 5.3), the choice picture matched the target picture. The participant was asked and ought to have made a tongue protrusion. So, this scenario tested for a true positive instance if the picture matching program detected a switch activation, and for a false negative instance if not. In a *mismatch* scenario (figure 5.4), the choice picture was not the target picture. The participant ought not to protrude his tongue. Therefore, this
scenario tested for a false positive instance if the program detected a switch activation, and for a true negative instance if not.

The values of $n_{TP}$, $n_{FN}$, $n_{TN}$, and $n_{FP}$ for each activity period could be calculated from the data collected by the picture matching program (see section 5.1.5) and by following the observations above. Denote the target picture index by $t$ and the choice vector by $\vec{c} = [c_1, c_2, \ldots, c_n]$, where the $c_i$’s are picture indices and $n = 20$. A scenario vector $\vec{s} = [s_1, s_2, \ldots, s_n]$ with elements in $\{0, 1\}$ was constructed by the rule

$$s_i = \begin{cases} 
1, & \text{if } c_i = t \text{ (i.e. matching scenario)} \\
0, & \text{otherwise (i.e. mismatch scenario).} 
\end{cases} \quad (5.3)$$

Denote the switch activation history vector by $\vec{h} = [h_1, h_2, \ldots, h_n]$. Recall that $\vec{h}$ also had elements in $\{0, 1\}$. Following the aforementioned observations exactly,

$$n_{TP} = \text{Cardinality of } \{i \in \{1, \ldots, n\} \mid s_i = 1 \text{ and } h_i = 1\}, \quad (5.4)$$
$$n_{FN} = \text{Cardinality of } \{i \in \{1, \ldots, n\} \mid s_i = 1 \text{ and } h_i = 0\}, \quad (5.5)$$
$$n_{TN} = \text{Cardinality of } \{i \in \{1, \ldots, n\} \mid s_i = 0 \text{ and } h_i = 0\}, \quad (5.6)$$
Chapter 5. Usability Experiment

Figure 5.4: An example of mismatch scenario.

(a) Target

(b) Choice

\[ n_{FP} = \text{Cardinality of } \{ i \in \{1, \ldots, n\} \mid s_i = 0 \text{ and } h_i = 1 \}. \quad (5.7) \]

If the participant behaved exactly as instructed during an activity period, then the scenario vector would have tracked the participant’s tongue protrusions exactly and so the four calculations above would be valid for this activity period. However, perfect behaviour was not always the case. Sometimes, the participant did not protrude his tongue as expected in a matching scenario because he was not aware of the match or he temporarily lost focus. Other times, the participant protruded his tongue during a mismatch scenario because he was being playful and misbehaved. If either or both of these two situations happened during an activity period, the aforementioned observations about the matching and mismatch scenarios did not apply for this period.

Therefore, the scenario vectors had to be corrected for imperfect participant behaviours first before using equations (5.4) – (5.7). Each scenario vector was cross-validated with the prototype log file (see section 5.1.5) from the same activity period. The choice markers in the log file aligned with elements of the scenario vector, i.e. the
the \( i^{th} \) choice marker associated with element \( s_i \) of the scenario vector. The two problematic situations warranted the following two corrections to the scenario vector:

- Set \( s_i = 0 \) if \( s_i = 1 \) and no tongue marker between the \( i^{th} \) and \((i + 1)^{th}\) choice markers.

- Set \( s_i = 1 \) if \( s_i = 0 \) and a tongue marker between the \( i^{th} \) and \((i + 1)^{th}\) choice markers.

After necessary corrections, the scenario vector truly mirrored the participant’s tongue protrusions.

The cross-validation step effectively decoupled the sensitivity and specificity metrics from the picture matching activity. Indeed, sensitivity and specificity had to be independent of the underlying single-switch activity because they were meant to measure the prototype’s performance towards tongue protrusions. Nevertheless, the picture matching activity was important to the sensitivity and specificity calculations in an implicit way. It decomposed each activity period into 20 time slots, and the time slots made it possible to count the number of true positives, false negatives, true negatives, and false positives.

The prototype log file had all the necessary fusion algorithm inputs to reproduce the switch activations for each activity period. It meant that sensitivity and specificity calculations for a single camera tongue switch were possible. The purpose of this single camera analysis was to investigate the utility of the peripheral cameras. It was done by discarding \( z_{\text{TMS}}^{(t)} \), \( z_{\text{FVM}}^{(t)} \), and \( z_{\text{Inhibit}}^{(t)} \) of the peripheral cameras. Only the fusion algorithm inputs of the center camera were extracted from the prototype log file and then fed into an offline implementation of the fusion algorithm. Recall that when there is only input from one camera, the fusion algorithm (see section 4.4) is functionally equivalent to the single camera switch activation (see section 4.3.3). Aligning the resulting fusion algorithm output with the choice markers yielded the center camera switch activation history vector \( \vec{h}_{cc} \) for the corresponding activity period. Finally, sensitivity and specificity
of the center-camera-only tongue switch for this activity period were calculated using the aforementioned procedure with $\vec{h}_{cc}$ replacing $\vec{h}$.

To further address utility of the peripheral cameras, data analysis included scrutiny of the true positive instances for the number of center camera switch activations versus the number of peripheral camera switch activations amongst these instances. First, reproduce the fusion algorithm calculations using the fusion algorithm inputs extracted from the prototype log file of the relevant activity period. Then, for each derived switch activation, back-tracing the inputs for the largest $z^{(t)}_{FWM}$ identified the camera that caused the switch activation. Finally, all derived switch activations were cross-referenced with the corresponding corrected scenario vector, retaining only switch activations that occurred during matching scenarios and, therefore, true positive instances.

### 5.3 Results

Table 5.1 summarizes data analysis of the five experiment sessions with the case study participant and the multiple camera tongue switch prototype. First, the mean sessional sensitivity is 82% and the mean sessional specificity is 80%. Second, the peripheral cameras are associated with most of the true positive switch activations in sessions 1, 2, and 5, and nearly half of them in session 4. Therefore, the peripheral cameras were indeed used to make many of the true positive switch activations.

Notice the larger standard deviations on the sessional average sensitivities than those on the sessional average specificities. This is mainly due to the fewer true positive instances and false negative instances in each activity period for estimating sensitivity, as compared to the number of true negative instances and false positive instances for estimating specificity. In particular, the sum $n_{TN} + n_{FP}$ dominates the sum $n_{TP} + n_{FN}$ in all five sessions by at least a factor of 3:1. The fewer true positive instances and

---

1 The mean sessional sensitivity and specificity are calculated from the five sessional average sensitivities and specificities, respectively.
false negative instances are consequences of the compromise for keeping a low number of matching scenarios in the choice vectors, in order to minimize participant fatigue during the experiment sessions.

Table 5.2 summarizes the center-camera-only analysis of the five experiment sessions. Table 5.3 compares the sessional average sensitivities and specificities between the multiple camera and the center-camera-only analyses. For all five sessions, the sessional average sensitivities decrease and the sessional average specificities increase for the center-camera-only case. The decreases in sensitivities are much larger for sessions 1, 2, and 5, where most of the true positive switch activations came from the peripheral cameras. For sessions 3 and 4, smaller changes were observed in sensitivity and specificity between the single and multiple camera cases. The center camera dominated the peripheral cameras during these two sessions and so we expect the center-camera-only case to perform as well as the multiple camera case.

The decreases in sessional average sensitivities and the increases in sessional average specificities for the center-camera-only case are significant observations. A decrease in sensitivity implies the tongue switch had more trouble reacting to tongue protrusions. Comparing the $n_{TP}$ and $n_{FN}$ columns of tables 5.1 and 5.2 reveals that the center-camera-only case missed more tongue protrusions. An increase in specificity implies the tongue switch committed less false positives, but the tongue switch would be naturally less susceptible to false positives if the tongue switch was already less responsive to tongue protrusions. In the extreme case, an unresponsive tongue switch that is inactive most of the time will artificially attain high specificity by virtue of inaction. Therefore, the center-camera-only configuration was actually less responsive to tongue protrusions, due to losing track of the mouth to the side view of the face. This means the peripheral cameras were actually necessary to capture the tongue protrusions that the center-camera-only case missed.
Table 5.1: Performance of the multiple camera tongue switch prototype. The session summary lines report the total counts and the sessional averages (and standard deviations) of sensitivities and specificities.

<table>
<thead>
<tr>
<th></th>
<th>Run</th>
<th>$n_{TP}$</th>
<th>$n_{TN}$</th>
<th>$n_{TP}$</th>
<th>$n_{TN}$</th>
<th>Sensitivity (%)</th>
<th>Specificity (%)</th>
<th>TP Activations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$n_{TP}$</td>
<td>$n_{TN}$</td>
<td>$n_{TP}$</td>
<td>$n_{TN}$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>14</td>
<td>1</td>
<td>80</td>
<td>93</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>10</td>
<td>4</td>
<td>67</td>
<td>71</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>16</td>
<td>1</td>
<td>67</td>
<td>94</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>0</td>
<td>14</td>
<td>0</td>
<td>100</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>17</td>
<td>1</td>
<td>100</td>
<td>94</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td>18</td>
<td>4</td>
<td>71</td>
<td>7</td>
<td>$83 \pm 17$</td>
<td>$91 \pm 11$</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>7</td>
<td>0</td>
<td>11</td>
<td>2</td>
<td>100</td>
<td>85</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>10</td>
<td>5</td>
<td>60</td>
<td>67</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>11</td>
<td>5</td>
<td>100</td>
<td>69</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>14</td>
<td>2</td>
<td>50</td>
<td>88</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td>16</td>
<td>4</td>
<td>46</td>
<td>14</td>
<td>$78 \pm 26$</td>
<td>$77 \pm 11$</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>12</td>
<td>2</td>
<td>83</td>
<td>86</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>11</td>
<td>5</td>
<td>50</td>
<td>69</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>10</td>
<td>6</td>
<td>100</td>
<td>63</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>14</td>
<td>3</td>
<td>100</td>
<td>82</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>14</td>
<td>3</td>
<td>100</td>
<td>82</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td>17</td>
<td>3</td>
<td>61</td>
<td>19</td>
<td>$87 \pm 22$</td>
<td>$76 \pm 10$</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>4</td>
<td>0</td>
<td>12</td>
<td>4</td>
<td>100</td>
<td>75</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>13</td>
<td>3</td>
<td>50</td>
<td>81</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>9</td>
<td>4</td>
<td>100</td>
<td>69</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>11</td>
<td>4</td>
<td>100</td>
<td>73</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>13</td>
<td>3</td>
<td>100</td>
<td>81</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>3</td>
<td>0</td>
<td>12</td>
<td>5</td>
<td>100</td>
<td>71</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>7</td>
<td>80</td>
<td>53</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>4</td>
<td>0</td>
<td>13</td>
<td>3</td>
<td>100</td>
<td>81</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td>33</td>
<td>3</td>
<td>91</td>
<td>33</td>
<td>$91 \pm 18$</td>
<td>$73 \pm 9$</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>14</td>
<td>3</td>
<td>67</td>
<td>82</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>16</td>
<td>1</td>
<td>67</td>
<td>94</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>33</td>
<td>93</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>13</td>
<td>3</td>
<td>100</td>
<td>81</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>4</td>
<td>80</td>
<td>73</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td></td>
<td>14</td>
<td>7</td>
<td>67</td>
<td>12</td>
<td>$69 \pm 24$</td>
<td>$85 \pm 9$</td>
<td>3</td>
</tr>
</tbody>
</table>
Table 5.2: Performance of the center-camera-only tongue switch prototype. The session summary lines report the total counts and the sessional averages (and standard deviations) of sensitivities and specificities.

<table>
<thead>
<tr>
<th>Run</th>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
<th>Sensitivity (%)</th>
<th>Specificity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0 5 15 0</td>
<td>1 6 13 0</td>
<td>5 1 18 0</td>
<td>4 0 15 2</td>
<td>1 4 12 4</td>
<td>0 100</td>
<td>67 100</td>
</tr>
<tr>
<td>2</td>
<td>4 2 9 5</td>
<td>2 3 15 0</td>
<td>0 4 16 0</td>
<td>1 3 15 1</td>
<td>6 1 11 2</td>
<td>67 64</td>
<td>40 100</td>
</tr>
<tr>
<td>3</td>
<td>2 1 17 0</td>
<td>1 2 15 1</td>
<td>5 0 11 4</td>
<td>4 0 15 1</td>
<td>2 2 16 0</td>
<td>60 93</td>
<td>50 94</td>
</tr>
<tr>
<td>4</td>
<td>0 6 14 0</td>
<td>0 4 16 0</td>
<td>0 3 10 5</td>
<td>0 3 10 5</td>
<td>0 4 14 2</td>
<td>0 100</td>
<td>0 94</td>
</tr>
<tr>
<td>5</td>
<td>1 1 18 0</td>
<td>2 2 15 1</td>
<td>6 1 16 1</td>
<td>4 2 15 1</td>
<td>1 4 13 2</td>
<td>20 87</td>
<td>33 94</td>
</tr>
<tr>
<td>Summary</td>
<td>7 15 73 5</td>
<td>1 6 13 0</td>
<td>5 1 18 0</td>
<td>4 0 15 2</td>
<td>1 4 12 4</td>
<td>37 ± 34</td>
<td>93 ± 16</td>
</tr>
<tr>
<td>1</td>
<td>1 6 13 0</td>
<td>14 100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 3 15 0</td>
<td>40 100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0 4 16 0</td>
<td>0 100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2 2 15 1</td>
<td>50 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>5 15 59 1</td>
<td>26 ± 23</td>
<td>98 ± 3</td>
<td>77 ± 25</td>
<td>86 ± 11</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5 1 12 2</td>
<td>83 86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2 2 11 5</td>
<td>60 69</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2 2 16 0</td>
<td>50 100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3 0 15 2</td>
<td>100 88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3 0 15 2</td>
<td>100 88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>15 5 69 11</td>
<td>77 ± 25</td>
<td>86 ± 11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4 0 12 4</td>
<td>100 75</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 3 15 1</td>
<td>25 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>6 1 11 2</td>
<td>86 85</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>5 0 11 4</td>
<td>100 73</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0 4 15 1</td>
<td>0 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1 2 16 1</td>
<td>33 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3 2 10 5</td>
<td>60 67</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4 0 14 2</td>
<td>100 88</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>24 12 104 20</td>
<td>63 ± 40</td>
<td>84 ± 11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>2 1 14 3</td>
<td>67 82</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0 3 17 0</td>
<td>0 100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0 6 13 1</td>
<td>0 93</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2 2 15 1</td>
<td>50 94</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1 4 13 2</td>
<td>20 87</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Summary</td>
<td>5 16 72 7</td>
<td>27 ± 30</td>
<td>91 ± 7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 5.3: Comparing the sensitivities and specificities of the tongue switch prototype across experiment sessions, using fusion algorithm inputs from all three cameras (multiple) versus only from the center camera (center-only). The bolded rows highlight the sessions where the peripheral cameras made a difference to the prototype’s performance.

<table>
<thead>
<tr>
<th></th>
<th>Sensitivity (%)</th>
<th>Specificity (%)</th>
<th>TP Activations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Multiple Center-Only</td>
<td>Multiple Center-Only</td>
<td>Center Peripheral</td>
</tr>
<tr>
<td>Session 1</td>
<td>83 ± 17 37 ± 34</td>
<td>91 ± 11 93 ± 16</td>
<td>5 13</td>
</tr>
<tr>
<td>Session 2</td>
<td>78 ± 26 26 ± 23</td>
<td>77 ± 11 98 ± 3</td>
<td>3 13</td>
</tr>
<tr>
<td>Session 5</td>
<td>69 ± 24 27 ± 30</td>
<td>85 ± 9 91 ± 7</td>
<td>3 11</td>
</tr>
<tr>
<td>Session 3</td>
<td>87 ± 22 77 ± 25</td>
<td>76 ± 10 86 ± 11</td>
<td>15 2</td>
</tr>
<tr>
<td>Session 4</td>
<td>91 ± 18 63 ± 40</td>
<td>73 ± 9 84 ± 11</td>
<td>19 14</td>
</tr>
</tbody>
</table>
Chapter 6

Discussion

6.1 General remarks on the tongue switch prototype

6.1.1 Average sensitivity and specificity

The multiple camera tongue switch prototype achieved an average sensitivity and specificity of 82% and 80%, respectively, towards intentional tongue protrusions by the participant across five usability experiment sessions. Considering that the participant was not fitted with an access technology prior to the case study, these results are promising with regards to prescribing the multiple camera tongue switch as the participant’s access technology. These results also show that it is possible to achieve good sensitivity and specificity with a non-contact tongue protrusion modality via video-based techniques in a controlled environment.

However, the usability experiment results cannot infer on the utility of the multiple camera tongue switch prototype for the participant for day-to-day single-switch access. This is because the picture matching activity as used in the usability experiment is not representative of the participant’s typical single-switch activities. The switch activations were made independently of each other in the picture matching activity, but this is not necessarily the case for more complex tasks that require inputs to be built up sequentially.
The tongue switch prototype is prone to false positives (see table 5.1), and the false activations will likely impede the construction of input sequences for more complex tasks.

### 6.1.2 Utility of the peripheral cameras

Results of the usability experiment confirm that the peripheral cameras are indeed necessary to capture tongue protrusions that the center camera otherwise would have missed. The usability experiment also verified that the peripheral cameras are actively contributing to the true positive switch activations.

It was hypothesized that the multiple camera tongue switch would be easier to use for an individual with severe spastic quadriplegia because he or she would not have to purposely target the center camera with the head. Observations of the participant during the five experiment sessions verified this claim. On the days where the peripheral cameras were used much more than the center camera for true positive switch activations (i.e. sessions 1, 2, and 5), the participant appeared to be more at ease not facing the center camera. Having his head turned away was his natural state on those days, and rotating his head in order to face the center camera would have required some effort. Because of his past experience with computers, the participant was already accustomed to reading the computer monitor with his neck laterally rotated and flexed.

### 6.1.3 Load on the processing unit

The processing unit of the tongue switch prototype (see section 4.2.3) takes approximately 25 ms to process one frame of video input from one camera. The dual core processor load ranges from 80% to 95% during typical operation. It is speculated that a quad core processor could host the multiple camera tongue switch and a single-switch application on the same computer.

The processor load is mostly influenced by the size of the bounding rectangle for the face region of interest. A larger face region of interest implies a larger search window
for CAMSHIFT and also more pixels to be analyzed by the mouth localization module. On the other hand, the size of the bounding rectangle for the mouth region of interest is usually too small to influence processor load.

### 6.1.4 Clinical advantages

There are some clinical advantages for realizing the tongue protrusion access modality via the tongue switch prototype. These advantages are consequences of the prototype’s non-contact property. Specifically, the non-contact property mitigates the risks of skin rashes associated with wearing sensors for an extended period of time. Furthermore, a non-contact tongue switch is more sanitary to use because it eliminates all direct oral contact between the user and the tongue switch. The tongue switch is easier to maintain with regards to cleanliness and the risks of infections are minimized.

### 6.2 Limitations of the tongue switch prototype

The multiple camera tongue switch prototype has the following technical limitations:

- *Skin-coloured objects occluding the user or in the backdrop can interfere with face localization.* Typically, distractors include exposed skin from other people and wood furniture (especially birch furniture). But they can also be non-skin objects in the environment that are retained by the skin-segmented frame (see section 4.3.1) due to coloured illumination or improper white balance. Regardless, if all the distractors are isolated from the user’s head, then the adaptive search window of the CAMSHIFT algorithm provides robustness against tracking the distractors (see section 4.3.1). The problem is with distractors near the user, where some pixels of the distractors are adjacent to the pixels of the user’s face and so CAMSHIFT tracks the distractors as part of the user’s face. A direct consequence is the skewing of the frontal view measure. Because the face region of interest is enlarged by distractors,
the assumption about the mouth being close to the left or right boundaries of
the face bounding rectangle when the head is turned to the sides may no longer
hold true (see section 4.3.3). Background subtraction with dynamic background
modelling [47] was attempted, to remove non-moving distractors from the skin-
segmented frame. This approach was eventually abandoned; either the non-moving
distractors did not converge into the background fast enough or the user’s head
became part of the background when the head remained stationary.

• Other saturated red colour patches inside the face region of interest can interfere
with mouth localization. Sometimes the lips and tongue are not the only satu-
rated red objects inside the face region of interest. Skin flushes introduce patches
of redness on the face that can interfere with mouth localization. A particularly
interesting episode was when the participant’s face flushed with excitement over
controlling a video game for the first time using the tongue switch prototype. How-
ever, such episodes should occur less frequently once the user becomes habituated
with access. Another source of saturated reds are skin irritations. This was partic-
ularly common at the ears of the participant, which rubbed against the headrest of
his wheelchair during spastic head movements. Therefore, the mouth localization
module sometimes tracked one of the ears instead. Third, shadows sometimes ac-
centuate the redness of skin regions. Finally, red clothing can dominate the redness
of the lips and tongue. This includes shirts, because the face region of interest
also contains the neck and so the bottom corners of the face bounding rectangle
capture shirt collars. Therefore, the user cannot wear red-coloured shirts or head
accessories while using the tongue switch prototype.

• The mouth localization module has difficulties with pale lips, thin lips, and facial
hair. Some individuals’ lips are not saturated in colour. These pale lips cause prob-
lems with mouth localization because the lip colours are not distinct enough from
skin tone. Other individuals have recessed lips. These cases are also problematic because there are too few lip pixels to perform connected component analysis and bounding rectangle aggregation (see section 4.3.2). Finally, moustaches and beards occlude the lips and thus interfere with mouth localization. Fortunately, this is unlikely a concern for young children.

- *The single camera algorithm has no provisions for changes to the environment illumination in real-time.* The single camera algorithm does not have automatic image conditioning. Instead, image conditioning is fixed by the camera exposure and white balance settings. These settings remain valid only if the environment illumination does not change significantly since the previous adjustments. For this reason, the single camera algorithm is not robust against real-time changes to environment illumination without reconfiguring the camera settings. This precludes the tongue switch prototype from being deployed near windows or outdoors, due to the sunlight fluctuations.

The tongue switch prototype has the following clinical limitations:

- *Manual configuration of the cameras is required to deploy the prototype.* The prototype is made easier to set up by a caregiver in terms of camera position and pose adjustments because of the independent camera configuration (see section 4.1). Still, the optics and colour balance of the individual cameras need to be tuned properly for the single camera algorithm. Although not a difficult process, the caregiver has to be taught how to tune the cameras. This also means the prototype has to be deployed by someone familiar with the system.

- *Deploying the camera mounting platform may not be possible in cluttered environments.* Mounting three cameras can be a challenge in some environments. The prototype’s mounting hardware (see section 4.2.2) has the benefit of holding the camera array in position. However, in cluttered environments such as a classroom,
there may not be enough room to place the platform. Also, it is a bulky platform to transport. Mounting the cameras on separate tripods allows for placement flexibility in cluttered spaces, but the individual cameras can be easily shifted out of position. Mounting the cameras onto the user’s wheelchair may provide both positional rigidity and system portability. Investigating alternative methods of camera mounting was beyond the scope of the case study.

6.3 Concerns with tongue protrusion as an access modality

6.3.1 Clinical concerns

Although tongue protrusions are predominantly voluntary movements, they are also rather infrequently performed by most individuals. The amount of tongue protrusions necessary for single-switch access far exceeds the amount of protrusions otherwise. This raises concerns from speech-language pathologists regarding the physiological development and well-being of oral and vocalization functions, especially with children. First, they have concerns with any long-term effects on the tongue muscles due to the frequent tongue protrusions. Second, they have concerns that the extra stress on the tongue muscles will cause the individual’s vocalization ability to deteriorate or will interfere with a child’s vocal development.

Appropriate responses to these clinical concerns are expected to differ from one case to the next. First and foremost, the child needs to feel physically comfortable with tongue protrusions, at least after some tongue exercising and training regime. Otherwise, tongue protrusion will not be a suitable access modality no matter how well the child can control his or her tongue. The concerns with vocalization and vocal development are contingent on the amount of residual vocal functions in the child. It is not uncommon to find indi-
individuals with severe cerebral palsy to also be non-verbal [13]. For these cases, the impact of repeated tongue protrusions on the child’s vocal development may be minimal. But for those children that have some vocal functions or potential for improved vocalization, it will be important to consider any trade-offs on vocalization before prescribing tongue protrusion for single-switch access.

Most importantly, the child needs to feel physically and psychologically comfortable with tongue protrusions as his or her access modality. Understandably, this may not be easy to ascertain in children with severe cerebral palsy. At the very least, the decision makers (i.e. caregivers and clinicians) should be aware of the possibility that the child may not share the same enthusiasm for the chosen modality.

6.3.2 Social acceptance

Unfortunately, tongue protrusions have some negative social connotations. In many cultures and social settings, protruding one’s tongue towards another person is inappropriate and can even be perceived as offensive. This is usually not an issue to the child’s caregivers or others who are familiar with the child. However, the tongue protrusions from a child using the multiple camera tongue switch are likely to be misinterpreted by strangers as impolite gestures directed towards them. The fact that the child will not be wearing any equipment, to hint at that he or she is using the multiple camera tongue switch, only adds to the likelihood of misinterpretation.

Another social risk is the transfer of tongue protruding behaviours to other social settings. This is particularly relevant for children. While the child is encouraged to practice tongue protrusions during training, the child also needs to be taught the social implications of this gesture. The child should be taught to understand that tongue protrusions are only appropriate when he or she uses the tongue switch. Otherwise, the child can be confused into believing tongue protrusions to be universally acceptable, as a result of the constant positive reinforcement received during training. Worse, the child
may protrude his or her tongue out of habit during other social encounters where such gesture is not appropriate.

Indeed, these social concerns are major reasons why some parents are reluctant to consider the tongue protrusion access modality for their child. Among the four respondents to our call for study (see section 3.2), the parents of two respondents expressed reservations about their children trying a tongue switch, partially due to concerns with social acceptance. However, these parents also indicated that they would consider the tongue protrusion modality for their children if the first attempts to fit an access solution were not successful. These comments suggest a greater willingness to accept and address the social concerns when the number of possible options for access declines.
Chapter 7

Conclusion

This research investigated a multiple camera approach of implementing non-contact and video-based access technology for children with severe spastic quadriplegic cerebral palsy. The research was conducted as a single descriptive case study of a 7 year-old participant with severe spastic quadriplegia (GMFCS level 5). The resulting access technology prototype was a multiple camera tongue switch. The choice of tongue protrusion was the outcome of an assessment session prior to the case study, which identified a facial gesture that could be harnessed for single-switch access.

7.1 Contributions

1. A colour video processing algorithm for a multiple camera tongue switch was developed.

The algorithm splits into two processing stages: a single camera stage and a fusion stage. Each camera has its own instance of a single camera algorithm. The video input of that camera goes through skin colour tracking for face localization, then saturated red colour feature analysis for mouth localization, and finally red pixels analysis and bounding rectangles analysis to derive a tongue-modulated statistic.
and a face frontal view measure respectively. The fusion algorithm sets the multiple camera statistic as the tongue-modulated statistic of the camera that has the highest frontal view measure. The multiple camera statistic is used in conjunction with positive edge thresholding to trigger switch activation.

There are two key characteristics to this multiple camera tongue switch algorithm. First, the colour video streams from the multiple cameras are treated as independent information sources. This allows for cameras to be added, removed, and repositioned at will. Second, the fusion algorithm follows a best camera paradigm; the camera with the best frontal view of the user’s face takes over the output of the multiple camera tongue switch.

2. Proof of concept of a non-contact tongue access modality.

In five sessions of a usability experiment with the case study participant in a controlled environment, the multiple camera tongue switch achieved an average sensitivity of 82% and specificity of 80%. These results show that it is possible to achieve good sensitivity and specificity with a non-contact tongue protrusion modality, via video-based method, for a child with severe spastic quadriplegia.

3. Demonstrated need for the peripheral cameras.

The usability experiment also verified the necessity of the peripheral cameras with regards to improving single-switch access. Specifically, across the five experiment sessions the peripheral cameras were responsible for many of the true positive switch activations. Furthermore, a “center-camera-only” analysis confirmed that the true positive switch activations by the peripheral cameras would have been missed by the center camera. Therefore, the peripheral cameras are necessary to cover those intentional tongue protrusions made by the user when his or her head is turned away due to head and neck spasticity.
7.2 Future work

The multiple camera tongue switch prototype should be deployed at the case study participant’s home or school to test for feasibility in the context of day-to-day single-switch access. The simple picture matching activity as used in the usability experiment is not representative of typical single-switch activities. Only by deploying the prototype for field operation will one be able to infer on the value of the multiple camera tongue switch for day-to-day single-switch access. This would be a natural next step to the case study of this research, or it could be the basis of a new case study with a different participant.
Bibliography

definition and classification of cerebral palsy, April 2005. Developmental Medicine

of body features to provide computer access for people with severe disabilities. IEEE


[5] Paola Campadelli, Raffaella Lanzarotti, and Giuseppe Lipori. Face localization in
color images with complex background. In Proceedings of the 7th International

and Te-Son Kuo. The new design of an infrared-controlled human-computer interface
for the disabled. IEEE Transactions on Rehabilitation Engineering, 7(4):474–481,
1999.


