Pervasive and Mobile Computing 10 (2014) 173–186

Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

Estimating and using absolute and relative viewing distance in interactive systems

Jakub Dostal *, Per Ola Kristensson, Aaron Quigley

School of Computer Science, University of St Andrews, UK

ARTICLE INFO

Article history:
Received 3 December 2011
Received in revised form 23 June 2012
Accepted 27 June 2012
Available online 8 July 2012

Keywords:
Viewing distance
Interaction
Mobile
Multimodal

ABSTRACT

In this paper we explore and validate the merits of using absolute and relative viewing distances from the screen as complementary input modalities for interactive systems. We motivate the use of viewing distance as a complementary modality by first mapping out its design space and then proposing several new applications that could benefit from it. We demonstrate that both absolute and relative viewing distance can be reliably estimated under controlled circumstances for both desktop and mobile devices using low-cost cameras and readily available computer vision algorithms. In our evaluations we find that viewing distance is a promising complementary input modality that can be reliably estimated using computer vision in environments with constant lighting. For environments with heterogeneous lighting conditions several challenges still exist when designing practical systems. To aid practitioners and researchers we conclude by highlighting several design implications for future systems.

1. Introduction

Researchers have overcome many of the robustness, initialisation, speed and usability issues that previously blocked vision-based techniques from evolving into mainstream user interfaces. Sensors such as Microsoft Kinect are used for body and gesture recognition, eye-tracking systems are used to create new accessible interfaces, motion tracking is integral to the film industry, and face detection and iris recognition are rapidly becoming commonplace in biometric security systems. Recognition of gaze and facial expressions has been researched extensively, see for example [1] for an extensive review.

In this paper we explore how we can use the viewing distance between the user and the display as a complementary interaction modality. Specifically, we make the following contributions:

- We map out the design space for techniques using viewing distance.
- We present an algorithm that can reliably estimate absolute and relative viewing distances ($\geq$90% accuracy) for both mobile and desktop devices under constant lighting conditions.
- We achieve these results with a markerless approach using readily available computer vision algorithms and commodity cameras.
- We investigate the accuracy of absolute and relative viewing distance estimation for both indoor and outdoor environments.
- We highlight several design implications for future systems.

* Corresponding author.
E-mail address: jd67@st-andrews.ac.uk (J. Dostal).

1574-1192/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2012.06.009

Table 1
Comparison of existing distance sensing methods.

System	Sensor position	Maximum detected distance
Our system	Environment	3 m+
Kinect [15]	Environment	Up to 5 m
Lean and zoom [11]	Environment (camera) + User (markers)	Unknown
Vicon	Environment (cameras) + User (markers)	Up to 12 m
OptiTrack	Environment (cameras) + User (markers)	Up to 15 m
InterSense IS-900	Environment (cameras) + User (trackers)	Up to 12 m
RFID+WiFi	Environment + User	Unknown

1.1. Related work

Researchers have previously explored a range of computer vision techniques for identifying and tracking facial features, including pupils, the eye area, nostrils, lips, lip corners and pose (e.g. [2–4]). Once detected and tracked, such features can form parts of a multimodal interface (for an extensive survey see [1]).

Face-tracking has been used to realise "perceptual interfaces" that allow face movement to control games [5] and 3D graphics interfaces [6]. Head movements can be translated into control variables, which can then be used to control a mouse pointer. However, under certain circumstances face-tracking using computer vision algorithms is insufficient to accurately control viewpoint movements. To overcome this limitation, researchers have proposed to fuse multiple modalities. For example, fused inputs from gaze, speech, mouse and keyboard have been explored to adapt the interfaces of office applications [7].

Researchers have also investigated how to detect and interpret user's gaze. Vertegaal et al. [8] use a custom *EyeContact* sensor coupled with a mobile phone to detect whether the user is engaging in a conversation. Later, Dickie et al. [9] propose a range of scenarios on how to use the same EyeContact sensor to adapt mobile applications depending on whether the user is looking at the mobile phone display or not. A related technique is to use gaze-gestures for mobile phone interaction [10].

In addition, several systems exist that measure users' proximity to their displays. Harrison and Dey [11] present a system that detects when users learn towards their laptop and then automatically zooms into the user interface. This system requires the user to wear markers for it to be able to estimate the user's distance to the display. In their study of ambient public displays, Vogel and Balakrishnan [12] use another marker-based approach, the Vicon motion-tracking system. They also define four zones of interaction: ambient, implicit, subtle and personal. This is a refinement of the three interaction zones (ambient, notification and cell interaction) defined by Prante et al. [13], who used a combination of RFID and WiFi sensing to determine distance. Ballendat et al. [14] expand the work of the previous papers by refining the interaction zones for proximity-aware interaction. Ballendat et al. [14] also use the marker-based Vicon motion tracking system.

There are a number of different technologies used for distance detection in the referenced work. We list the available technologies (or their current alternatives where the product has been superseded by a newer version) in Table 1. We observe three distinct groups of sensors.

- High accuracy, long range sensors that require user augmentation.
- Sensors with lower accuracy and range requiring no user augmentation.
- Other systems.

The high range and accuracy group includes two marker based systems using a number of cameras in the environment and passive markers placed on the object to be tracked (Vicon, OptiTrack). There is also an ultrasonic system (InterSense), which uses tracker devices worn or held by the user in addition to the sensors placed in the environment. The Vicon system has been used in several referenced works [12,14] and the InterSense system was used for example by Nacenta et al. in their study of perspective-aware multi-display interfaces [16].

The main advantage of the second group of sensors is that they require no augmentation of the user. This group includes our system and the Kinect sensor. Where our system uses standard computer vision algorithms, the Kinect sensor has a special camera capable of capturing depth information. The maximum detection distance of our system has not been established yet as it depends on the resolution of the camera available to the system. It also means that as camera sensor technology improves, the maximum detection distance of our system will improve as well.

The third group consists of sensor systems that do not fit into the groups above. They both need to augment the user to work. The infrastructure based system used by Prante et al. [13] has only been specified very loosely in their paper but includes a Pocket PC held by the user, and a WiFi device and an RFID reader present in the environment. The system described by Harrison and Dey [11] uses a similar approach to ours but they use markers on the user's head to determine the distance.

1.2. Viewing distance as a complementary modality

In this paper we propose viewing distance as a complementary modality. Such a modality can be useful for a variety of applications. For example, it can enable mobile systems to detect whether the user is looking at the screen or not

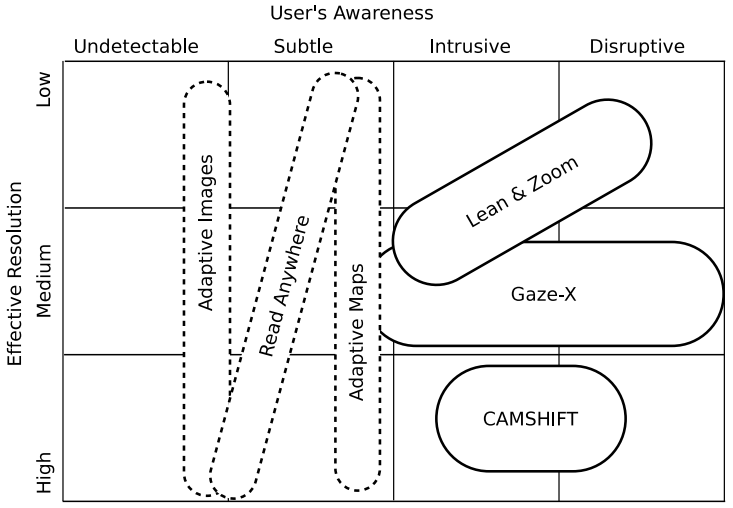


Fig. 1. Design space of systems that could use viewing distance for interaction.

(e.g. [9]). It can also drive a system that automatically zooms in when users are leaning towards the screen (e.g. [11]). In addition, this modality can be useful to automatically dismiss notifications when the user is looking at their mobile phone or desktop screen.

In Fig. 1 we map out the design space of viewing distance in two dimensions: effective resolution and human detection level. In the figure, the human detection level ranges from undetectable to disruptive. Effective resolution is a combination of the viewing distance (and thus the viewing angle) of the user, the pixel resolution of a display, and the physical dimensions of the display. We give examples of three potential applications that map to the three most common types of content presented on displays: images, text and visualisations.

One design concept is to use viewing distance to create a system for adaptive viewing of images. Such a system would determine the ideal resolution of an image based on the distance between the user's eyes and the screen. This interface design would lie between undetectable and subtle as it would be hard for the user to notice that the resolution of downloaded images depended on the distance between their eyes and the screen. This is because there would always be enough detail for the pixels of the displayed image to be slightly beyond the resolving capability of the user.

Another example is an adaptive mapping system that would display varying amounts of detail in the maps depending on the viewing distance of the user. This system is on the boundary between subtle and intrusive. On the one hand, the changes in the amount of detail shown would be noticeable. On the other hand, the interface would appear less cluttered.

The third design idea is to dynamically adapt textual information. Consider a resolution-independent e-book reader. No matter how far away the user is from the display, the system can ensure the text is shown at a constant size from the user's point of view. In this design the user can select the most comfortable font size for the text and the reader can automatically adjust the amount of text shown based on the available display area (which is proportional to the viewing angle). We hypothesise that this system would reside in the subtle range. Although the changes in the amount of text shown will be noticeable, they would probably not distract the user much from their reading task.

Fig. 1 also maps out three other existing systems. Bradski's CAMSHIFT head-tracking system [6] is used as a control interface for games and spatial navigation in 3D environments. Since the head movement directly controls all the movement within the interface, the changes in the user interface are intrusive, bordering on disruptive. Harrison and Dey [11] present a system that magnifies content on screen as the user leans in towards the display. Although the changes are still very noticeable, they are somewhat less pronounced than in the CAMSHIFT system. The third system is the Gaze-X system [7]. While it does not use viewing distance to alter the user interface, some of the described alterations of their affective user interface system could be motivated and directly enhanced using viewing distance as an input modality. The system has the widest variety of potential changes to the user interface, ranging from disruptive to almost subtle, providing their system with more flexibility compared to the other systems above.

2. Estimating viewing distance

In order to use viewing distance as a complementary modality we need to be able to measure it accurately. Preferably we can do so using inexpensive commodity hardware. Previously, the most closely related systems for measuring viewing distance, or detecting whether the user is looking at the screen, have relied on custom hardware [9] or markers [11]. In contrast, we explore a system designed to work with commodity hardware.

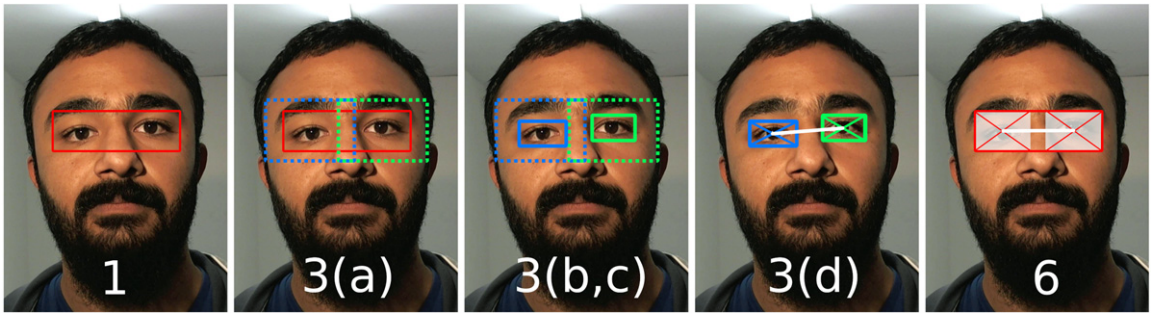


Fig. 2. A visual representation of the different stages of the distance detection algorithm. The numbering of the images corresponds to the stages of detection.

Our system uses the OpenCV computer vision library.[1] This decision was based on an evaluation by Castrillón-Santana et al. [17] who use OpenCV to perform feature detection for facial detection. We use OpenCV to detect faces and eyes for distance estimation. The feature detection algorithm in OpenCV is an implementation of the Viola–Jones feature detection algorithm [18].

The two different classifiers used in this paper were chosen based on observations of the available raw data, the data needed for distance estimation, and on the required processing time. As we will see they each have their own strengths and weaknesses in different experimental settings.

The One-Stage classifier uses a Haar cascade from the OpenCV library trained to detect eye-pairs (this classifier is referred to as EP1 in [17]). The classifier provides has two advantages. First, it reduces the risk of recognition errors as the eyes in conjunction with the nose form a more complex, and therefore more visually distinct object than the eyes do on their own. Second, the Haar cascade is trained primarily on frontal face images. Because of this the eye-pair is not detected when the user is not directly facing the camera (and, by extension, when the user is not looking at the screen).

The Two-Stage classifier uses the same underlying technique as the One-Stage classifier. However, instead of using a single cascade it uses two. The first cascade detects faces (this cascade is referred to as FAT in [17]). The area of the image where the face is located is then used as an input area for the second cascade. The second cascade is the same as in the One-Stage classifier. We test the Two-Stage classifier because the One-Stage classifier sometimes results in incorrect classifications. By ensuring that we only look for the eye-pair in an area where eyes are expected, the risk of recognition errors can be reduced. The disadvantage of this two-stage approach is that it is more resource-intensive.

We set the parameters of the OpenCV function for detecting objects using a Haar cascade as follows. First, the scale factor is 1.2, which means the search window size is increased by 20% between scans of the image. Second, the number of neighbouring rectangles that make up an object were set to one. This means that groups of less than one rectangle are rejected. Third, the system uses a Canny edge detector to reject image regions that contain too many or too few edges and thus cannot contain the object being detected. The net result of this is a speed increase.

2.1. Algorithm for estimating absolute viewing distance

The absolute distance estimation algorithm consists of two parts: image analysis and distance computation. The algorithm for image analysis is based on the One-Stage algorithm with some additional steps to increase accuracy. The main steps of the algorithm are visually illustrated in Fig. 2.

For a given image, the algorithm works as follows.

1. Find the areas of interest (the rectangles containing individual eye-pairs) using the One-Stage/EP1 classifier.
2. If no eye-pairs are found, fail and take no action.
3. Otherwise, for each detected eye-pair:
   (a) Partition the area of interest horizontally into two parts, adding 20% horizontal and 60% vertical padding to each of the two sub-areas of interest.
   (b) Scan the left sub-area of interest for potential rectangles containing the left eye using the LE classifier and select the largest one.
   (c) Scan the right sub-area of interest for potential rectangles containing the right eye using the RE classifier and select the largest one.

[1] http://opencv.willowgarage.com.

(d) If both sub-areas of interest contain a successfully detected eye, find the position of the pupils. Each pupil is located in the centre of the containing rectangle for its respective eye. Then, compute the viewing distance according to the formula below.
(e) Store the location of the pupils and the sizes of the containing rectangles for the eyes together with the computed distance.

4. From all the possible eye-pairs, select the one that has both eyes detected and for which the sizes of the rectangles containing each eye closely match each other.
5. If such an eye-pair is found, report a full detection, including the distance between the pupils within the area of interest in pixels.
6. If no such eye-pair is found, fail gracefully by noting a partial detection with the location of the largest eye-pair found as the most likely eye-pair. This is computed using the same method but instead of using the position of the pupils, we partition the rectangle occupied by the eye-pair into three horizontal segments (40%, 20%, 40%—see the last image in Fig. 2 for an example) and use the centres of the two 40% segments as the distance points.

The distance is computed based on the following formula:

$$D = \frac{\frac{D_e}{2}}{\tan\left(\frac{D_p}{2} \times \frac{C_a}{C_r}\right)}$$

where D—the distance of participant's eyes from the camera in millimetres
D_e—participant's pupil distance in millimetres (measured during calibration)
D_p—pupil distance in pixels (as computed in step 2(d) of the algorithm)
C_a—camera's horizontal angle of view in degrees
C_r—camera's horizontal resolution in pixels.

In its current form the algorithm is strict in its processing and only proceeds to distance computation when the most precise position of the individual eyes is available. However, under some circumstances it may be beneficial to provide a rough distance estimate even when more precise data is unavailable. In that case, the algorithm could fail gracefully by computing the distance based on the eye-pair rectangle rather than specific eyes. While this will be inherently less precise, the precision can be sufficient for a rough estimate. This graceful fallback is demonstrated in the last image in Fig. 2.

3. Evaluation 1: gaze detection and gaze direction detection

In the first evaluation we tested the system's performance in a controlled laboratory setting. We examined two input devices: a desktop computer equipped with a screen with a built-in web camera (Desktop) and a mobile phone with a front-facing camera (Mobile).

3.1. Method

In the experiment we investigated two settings (Desktop and Mobile), two classifiers (One-Stage and Two-Stage), and two viewing distances.

For the experiment we recruited four participants from our university campus. Their ages ranged between 23 and 29. All were male. Two participants wore glasses while the other two used neither glasses nor contact lenses.

The Desktop setting used an iMac 11,3 with an integrated web camera. The Mobile setting used an iPhone 3GS and its integrated camera. Since we aimed for consistency between the settings and the iPhone's camera was limited to VGA resolution (640 × 480 pixels), we used the same resolution for both of the settings.

For the Desktop setting the viewing distances were 90 cm and 60 cm. These distances approximated to distances at which the participants were comfortable viewing the screen. The distances were equivalent to horizontal viewing angles of 36° and 53°, respectively.

For the Mobile setting the viewing distances were 60 and 30 cm. 60 cm approximately corresponded to holding the mobile phone at an arm's length, while 30 cm corresponded to holding the mobile phone at a closer and more comfortable distance.

Markers were positioned around the room to provide participants with visual targets to make the gaze directions as consistent between participants as possible. They were all also advised to move their head, not just their eyes, as the important factor was not the movement of the eyeballs but rather the movement of the head, as this changes the visual appearance of the eye-pair the most from the point-of-view of the camera.

We positioned four markers surrounding the participant. Fig. 3 shows a diagram for how a prototypical marker is positioned. Point A was the middle of the participant's nose. In the Desktop setting point B was the centre of the screen. In the Mobile setting point B was the centre of the mobile phone screen. Point C was the position of the marker. In the Desktop setting the distance to the screen was 75 cm. The angle α was 30° for up, 60° for down, and 70° for left and right. In the Mobile setting the distance to the screen was 45 cm. The angle α was 60° for up and down, and 70° for left and right.

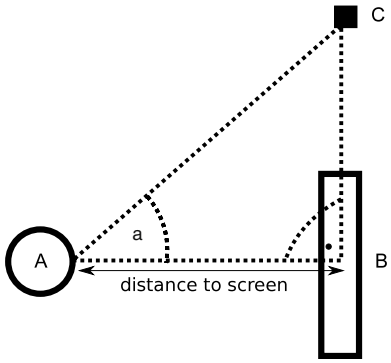


Fig. 3. A diagram illustrating the positioning of the user (A), the display (B) and one of the targets (C) for evaluation 1.

Table 2
Comparison of accuracy of classifiers. In order to provide more granularity to the data, we also include the percentages of true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN).

Setting	Classifier	Distance	Accuracy (%)	TP (%)	TN (%)	FP (%)	FN (%)
Mobile	One-Stage	Near	67.08	43.25	23.84	18.17	14.74
		Far	72.79	48.95	23.84	14.77	12.44
	Two-Stage	Near	68.69	27.76	40.94	0.88	30.43
		Far	62.91	27.41	35.49	6.02	31.07
Desktop	One-Stage	Near	76.87	56.49	20.38	22.94	0.19
		Far	72.96	55.08	17.88	26.30	0.74
	Two-Stage	Near	82.54	54.50	28.04	15.30	2.17
		Far	69.74	47.47	22.26	21.48	8.79

In each of the four different configurations, we instructed each participant to look for ten seconds at each direction determined according to the following sequence: centre, left, centre, up, centre, right, centre, down, centre. This means that participants looked directly at the screen 55% of the time. The other 45% of data enabled us to find out how well the classifiers handled different gaze directions.

3.2. Results

In total, we collected 24 min of video data. For each participant we collected 90 s of video data for each configuration (six minutes in total). Every video frame was proportionally scaled down to 66% of its original size in order to save processing power during collection.

The video stream was then processed by the One-Stage and Two-Stage classifiers to generate information about the eye-pairs they detected. The video stream was sampled at 30 Hz and the classifiers processed every single video frame sampled.

We saved the region that enclosed the eye-pairs whenever the classifiers detected them. The positions and sizes of the bounding boxes of every single eye-pair were also saved to form the basis of a distance profile. All of the saved data was timestamped so that each of the samples could be easily identified and synchronised with any other data.

A human judge marked the image data outputted by the classifiers into one of three categories:

- Valid: the classifier believed that the participant was looking at the screen and the human judge agreed.
- Edge case: the classifier believed that the participant was looking at the screen and the human judge disagreed. However, the eye-pair was captured properly. It was only the direction of the gaze that was wrong.
- Invalid: a complete misclassification. The classifier believed the participant was looking at the screen, when in fact there were no human eyes present in the sample image.

Each video stream was then coded by a human judge with information about the start and end times of the experiment and information about whether or not the participant was looking at the screen. The same judge also marked the parts of the stream, where it was impossible to tell where the participant was looking. Additional information about gaze direction was encoded as well to allow an analysis of how well the classifiers handle the different gaze directions.

Table 2 shows how accurate the classifiers were in detecting whether the participants were looking at the screen or not. Overall, the One-Stage classifier performed better in the Mobile setting than the Two-Stage classifier. The Two-Stage classifier performed better in the Desktop setting, but not by a large margin.

Table 3 shows how the classifiers handled different gaze directions. In the Mobile setting the One-Stage classifier had an accuracy higher than 90% along the lateral axis (centre, left, right). However, the accuracy for up and down movements

Table 3
Comparison of accuracy of classifiers for gaze direction.

Classifier	Centre (%)	Up (%)	Down (%)	Left (%)	Right (%)
Mobile environment					
One-Stage	91.06	37.40	0.00	95.00	92.42
Two-Stage	49.95	96.70	69.08	98.46	99.04
Desktop environment					
One-Stage	100.00	0.00	4.67	94.93	76.42
Two-Stage	90.81	0.00	48.29	95.93	87.12

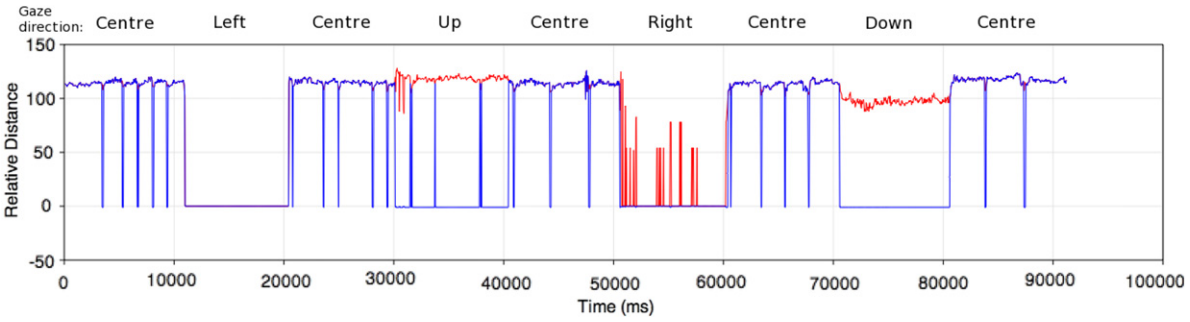


Fig. 4. A comparison of valid and invalid classifications in the Desktop setting for a typical participant. A perfect classifier would follow the blue line. The occasional dips during periods when we expected detections correspond to the participant blinking and thus were not considered to be incorrectly classified. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

was low (see the second and third columns in Table 3). The Two-Stage classifier performed notably better in every single gaze direction except for the centre. This was because the Two-Stage classifier tended to be overly conservative because it had difficulties in detecting faces at close distances.

In the Desktop setting the overall performance of the classifiers was similar for all gaze directions, except when participants were looking down. The Desktop setting faired much worse in detecting up and down movements than the Mobile setting. We hypothesise this is due to the angular difference between the centre point of the screen and the specific markers participants were gazing at when looking up or down being less for Desktop than Mobile.

Fig. 4 demonstrates the difference between correctly and incorrectly classified data. It was produced by the One-Stage classifier. The blue line shows how a perfect classifier would classify the data. The red line shows the actual classification outcome. If the classifier is always accurate the red line will always be hidden behind the blue line. The red line is only visible in the figure when the classifier has made a classification error.

The experimental sequence can be easily seen in the figure. Each of the approximately 10,000 ms segments shows different directions of gaze (as labelled above the data). The figure shows that for the centre and left gaze directions, the classifier performs very well. However, for small parts of the right direction segment and all of the up and down segments, the classifier completely misclassified the data.

4. Evaluation 2: real life eye-pair detection

The first evaluation examined our ability to detect the presence of gaze in a controlled environment by detecting the participant's eye-pair. In this second evaluation we wanted to test how well our classifiers would work under more real-life conditions through a task involving a mobile phone in a combination of outdoor and indoor environments.

4.1. Method

The scenario for this evaluation was to simulate a person walking somewhere while simultaneously trying to perform a task on a mobile phone.

Sudoku was chosen as the task because most people are familiar with the game, the rules are simple, and it does not require any complex mathematical knowledge in order for the user to be able to play it. Moreover, it is independent of language and culture, which means that we can discount any linguistic or cultural influence on the participant's ability to complete the task. The task is also time consuming enough to last the whole length of the experiment, and it can be easily repeated in case more time is needed for data collection.

Seven participants were recruited for this experiment, their ages ranged between 19 and 29. Four of the participants were female, three male. Four participants wore glasses and the other three participants wore neither glasses nor contact lenses.

Fig. 5. Simulating a front-facing camera.

We used the same mobile phone as in the first evaluation to collect our data. We attached two separate mobile devices to simulate a mobile phone with a front-facing camera. The iPhone was turned and connected to an iPod Touch with an offset so that the camera on the iPhone was facing the participant. The screen of the iPhone was covered in order to ensure the touch capability was disabled to avoid accidental touch input. Fig. 5 shows the resulting mobile device. The resulting device was somewhat less firm and easy to hold than a single device would have been. However, none of the participants reported any trouble in using it.

First, each participant familiarised themselves with the equipment and demonstrated that they knew how to play Sudoku. Thereafter they were taken on an approximately eight minute journey around an indoor laboratory environment and a surrounding outdoor area. On the journey, they followed a guide. The guide ensured that every participant would follow the same path and at a similar pace.

The path presented the participants with a range of environments (indoor, outdoor) and obstacles (navigating around furniture, opening/closing doors, ascending/descending stairs). It also exposed the equipment to a range of lighting conditions (uneven artificial lights, clear/cloudy sky outdoors).

4.2. Results

Each of the participants generated about eight minutes of data. In total we collected nearly an hour of video. Every video frame was proportionally scaled down to 66% of its original size in order to save processing power during collection.

The video stream was processed by the One-Stage and Two-Stage classifiers to generate the same type of output data as in Evaluation 1. However, the sample rate was reduced to 10 Hz in order to reduce the number of samples for processing. The image data was marked by the human judge into the same classes as in Evaluation 1.

Video stream segments were annotated into three classes: (a) participant looking at the screen, (b) participant not looking at the screen, or (c) insufficient visible data to determine whether the participant was looking at the screen or not. An example of an instance of the latter class is if the face is so dark that the eyes cannot be distinguished.

The One-Stage classifier resulted in a much higher accuracy (46.1%) than the Two-Stage classifier (16.8%). This is because the Two-Stage classifier often failed to detect the face. This resulted in an immediate classification failure. However, note that the failure to detect the face was often caused by the fact that the full face of the participant was often not present in the frames due to the narrow field of view of the camera and the participant's closeness to the camera.

Fig. 6 shows a representative data segment for the best-performing One-Stage classifier for a typical participant. The red line shows relative distance as it was detected by the One-Stage classifier. The blue line is a binary function that shows ground truth about whether the participant was looking at the screen or not (shown as values of either 250 or zero respectively). As is evident in the figure, even though this classifier performed reasonably well, in comparison to the results in Fig. 4, the results in Evaluation 2 are notably worse.

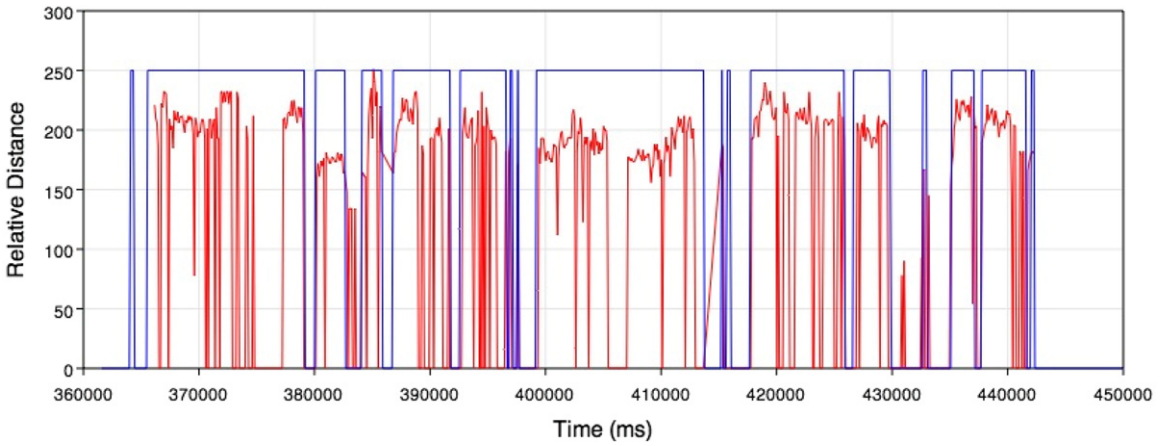


Fig. 6. A segment of a distance profile for a typical participant in Evaluation 2.

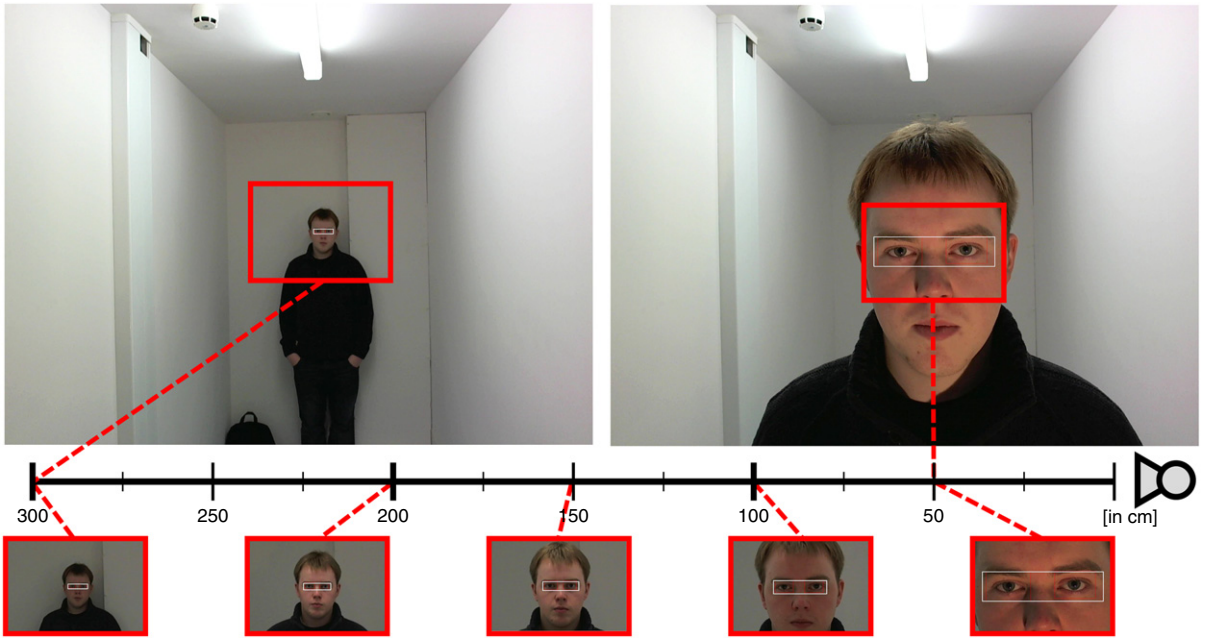


Fig. 7. An illustration of the difference in eye sizes with distance changes. All images in red rectangles are 100% crops. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

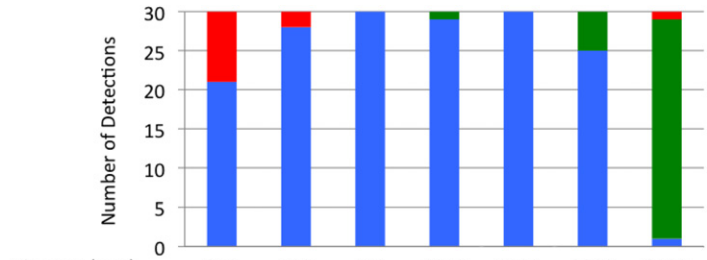
5. Evaluation 3: estimating absolute viewing distance

The third evaluation investigates how well we can estimate absolute viewing distance using the algorithm presented earlier in this paper.

5.1. Method

We recruited 15 participants. Their ages ranged between 19 and 26 (mean age 21.93). Three participants were female and twelve were male. None of the participants wore glasses.

Seven distances were chosen: 25, 50, 75, 100, 150, 200 and 300 cm. Distances up to 100 cm are more fine grained as they are more representative of the distances common for interacting with mobile phones and desktop computers. Distances between 100 and 300 cm are less fine grained for two reasons. First, these are less common interaction distances in desk-based systems. Second, the ability of a computer vision algorithm to detect an object in an image depends on the size of the object in pixels. With increasing distance, the size of the eyes will decrease geometrically (see Fig. 7). This means that the



Distance (mm)	250	500	750	1000	1500	2000	3000
No Detection	9	2	0	0	0	0	1
Partial Detection	0	0	0	1	0	5	28
Full Detection	21	28	30	29	30	25	1

Fig. 8. Distance detections at medium resolution (1600 × 896 pixels).

difference in size is less pronounced at longer distances and therefore close distance intervals would not contribute much to the results.

A consumer-grade webcam (Logitech C910) was mounted on a tripod and the chosen distances were measured from the position of the camera in a straight line and marked up on the floor. Since computer vision algorithms tend to be influenced by lighting conditions, a desk lamp was positioned so that it would provide additional light for short distances as the ambient lighting was rather dim (only one fluorescent light on the ceiling).

Due to the fact that the angle of view of the camera changed depending on the chosen resolution, two different resolutions were tested: a 4:3 ratio high resolution (2592 × 1944) and a 25:14 ratio medium resolution (1600 × 896). The horizontal angle of view of the camera was 62° for the standard 4:3 ratio resolution and 70° for the wide 25:14 ratio resolution.

During the evaluation, each participant was positioned on each of the distance marks on the floor in turn. A ruler positioned perpendicular to the floor was used to ascertain that the eyes of the participant were positioned above the distance mark. Two still pictures from the live camera video were captured. After the capture, the participant would be led to the next distance mark on the floor. When all the still pictures for the seven distances had been captured, the procedure was repeated for the second resolution.

5.2. Results

A total of 420 still images were collected for this evaluation (two captures for each of the seven distances at two different resolutions for each of the fifteen participants).

For some of the captures, the auto-focusing mechanism of the webcam failed to focus properly (four captures). This resulted in some extra softness in most images captured at 1600 × 896. The softness was probably due to in-camera interpolation for wide-ratio resolutions as we observed softness at other wide-ratio resolutions as well. Due to the lighting setup in the room, many of the long distance captures (at 300 cm) did not have much light illuminating the participants. For short distance captures, the secondary light source sometimes resulted in areas of shadows on the faces of some participants.

We also found four instances of a participant blinking and 24 instances of a participant not looking perfectly straight at the camera. However, we decided to include all the captures in the processing rather than excluding them as they better reflect real-life situations and conditions that our system is likely to encounter in use.

Figs. 8 and 9 show that the algorithm very rarely failed to completely identify the eyes (5.71% for 1600 × 896 and 4.76% for 2592 × 1944 over all the distances). Failed detections arose at very short distances (25 cm or 50 cm), or at the maximum distance (300 cm).

Partial detections only occurred at the lower resolution (1600 × 896) and at long distances (200 and 300 cm). This indicates that at long distances the lower resolution provides sufficient information to detect the eye-pair, but insufficient information to determine the accurate position of each individual eye.

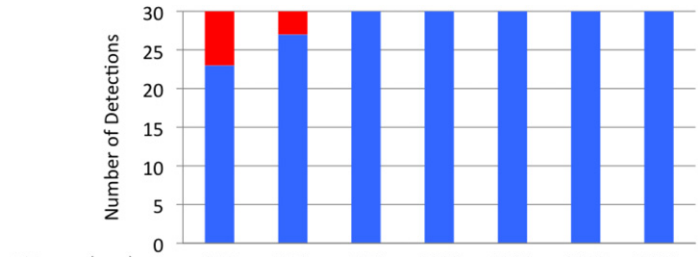
Figs. 10 and 11 show the accuracy of distance estimations for all the complete detections at the different distances. Except for the distance at 300 cm at the 1600 × 896 resolution, the estimation for each distance is based on at least 21 captures, and in most cases almost 30 captures.

The mean estimated distances have a relative error of no more than 11% at worst.[2]

The worst estimation error for the lower resolution was an error of 46 cm at a distance of 200 cm. The worst estimation error for the higher resolution was 36 cm at a distance of 300 cm. In general, estimation errors increase with distance as the pixel resolution becomes more important as the distance increases.[3]

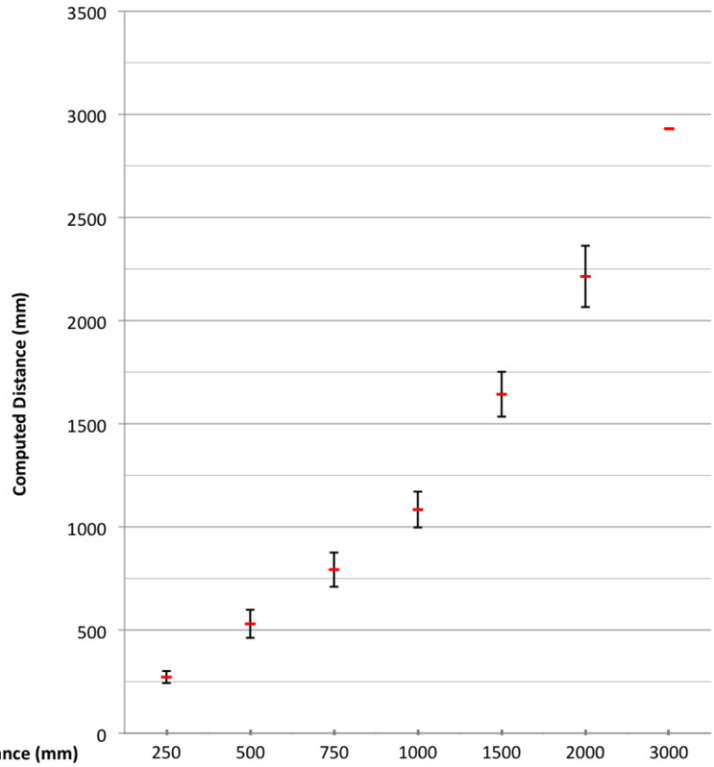
[2] The relative error is the ratio between the estimation error and the actual distance.

[3] At 300 cm, a pupil distance of 63 mm corresponds to approximately 50 pixels for a horizontal resolution of 2592 pixels and only 27 pixels for a horizontal resolution of 1600 pixels.



Distance (mm)	250	500	750	1000	1500	2000	3000
No Detection	7	3	0	0	0	0	0
Partial Detection	0	0	0	0	0	0	0
Full Detection	23	27	30	30	30	30	30

Fig. 9. Distance detections at high resolution (2592 × 1944 pixels).



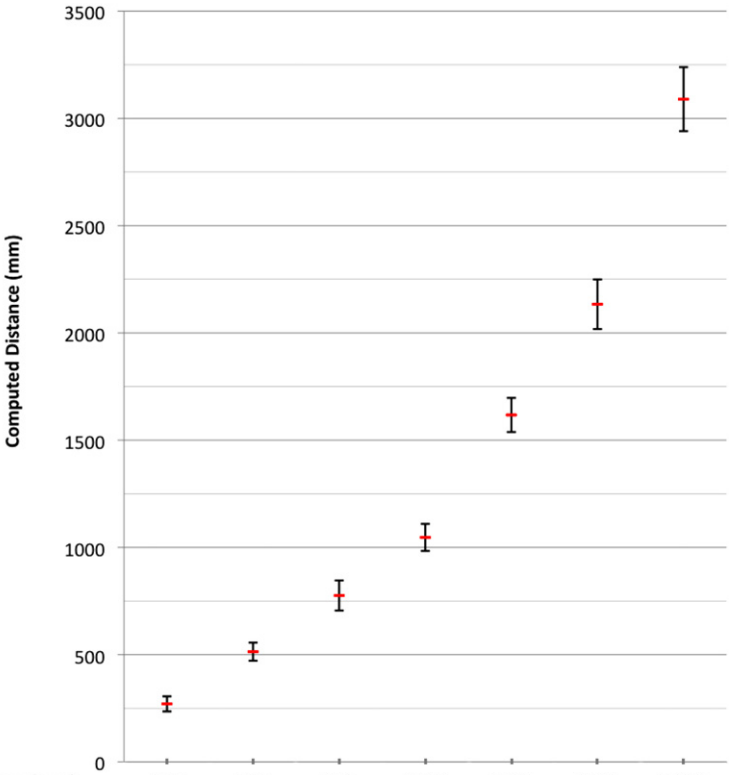
Expected Distance (mm)	250	500	750	1000	1500	2000	3000
Maximum Computed Distance	327	801	1052	1348	1921	2460	2930
Minimum Computed Distance	219	464	689	950	1410	1902	2930
Average Computed Distance	272	530	793	1084	1643	2214	2930

Fig. 10. Results of absolute distance evaluation at medium resolution (1600 × 896 pixels). Error bars show standard deviation.

6. Discussion

Our evaluations revealed that it is indeed possible to detect viewing distance using built-in and consumer grade cameras on desktops and mobile phones. The first evaluation showed that in a controlled environment, relative distance can be accurately estimated using computer vision. The third experiment further demonstrated that in a controlled environment with constant lighting conditions we can estimate users' absolute viewing distance with high accuracy.

However, the second evaluation demonstrated that estimation of relative viewing distance decreases dramatically in accuracy in heterogeneous use-scenarios with varying lighting conditions. When participants were walking around in the second evaluation, some participants positioned the camera so that their faces were not fully contained in the camera image. This resulted in classification failures, in particular for the Two-Stage classifier. Cameras with wide-angle lenses



Expected Distance (mm)	250	500	750	1000	1500	2000	3000
Maximum Computed Distance	350	641	986	1167	1811	2319	3364
Minimum Computed Distance	217	470	704	935	1476	1885	2808
Average Computed Distance	271	514	776	1047	1617	2134	3090

Fig. 11. Results of absolute distance evaluation at high resolution images (2592 × 1944 pixels). Error bars show standard deviation.

could potentially reduce this problem. Also, some participants used glasses while others did not. At certain angles glasses caused reflections that were seen by the camera in such a way that the reflection concealed their eyes and thus caused misclassifications. The rims of the glasses sometimes also obscured the eyes. Last, a minor issue we observed was that in some cases participants' long hair sometimes obscured their faces or eyes.

6.1. Dynamic range and exposure control

While the problems listed until now caused misclassifications, there is only a limited amount of what can be done as users will be using glasses and some will have long hair. We have already mentioned the lens of a camera as being a potential issue in specific circumstances.

However, a more serious issue is the limited dynamic range of the camera sensor. The dynamic range is the interval between the minimum and maximum light intensity that the camera can sense. The limited dynamic range of today's consumer sensors means that high contrast scenes cannot be completely accurately captured. Due to the fact that the exposure algorithms of the camera tried to balance the scene, the faces of the participants often became completely black and lost all distinguishing features. For example, the far left picture in Fig. 12 shows a frame from an outdoors segment in evaluation 2. As is evident in the figure, the participant's face is captured without any distinguishing features. This makes automatic eye detection via the camera impossible. This is a hardware problem we anticipate will be lessened as consumer-grade camera technologies continue to improve.

However, given this limitation of current consumer-grade camera hardware it becomes crucial to be able to programmatically set correct exposure settings for the camera. This is because if we have even a single correctly exposed frame we can accurately detect the user's eyes. Using such past recognition results, we can determine "an area of interest" that surrounds the known prior locations of the eyes. If mobile phones and desktop computers enabled programmatic changes of exposure settings we would be able to tell the camera to focus and meter around this "area of interest". This would mitigate the issue with limited dynamic range. To demonstrate this, the second picture from the left in Fig. 12 shows the resulting frame produced by a Logitech C910 camera's automated exposure algorithms. The third picture from the left in Fig. 12 shows a frame produced by manual correction of exposure. This latter frame captures the face with enough detail

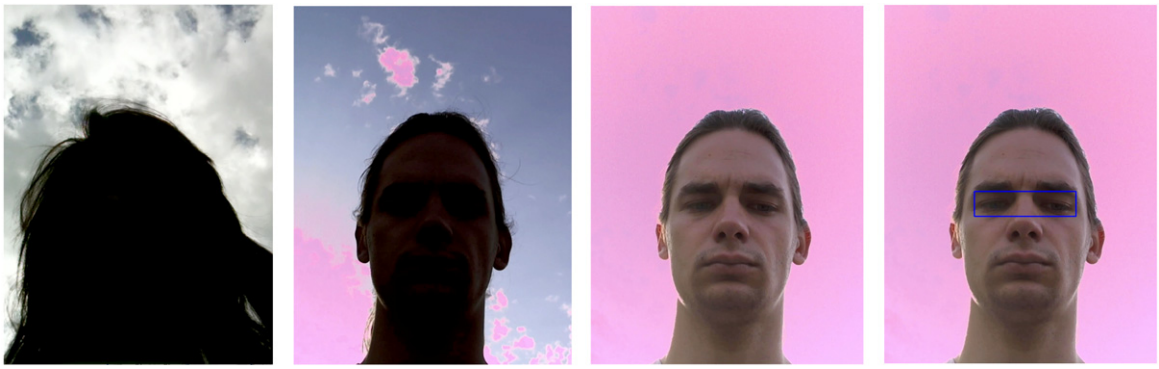


Fig. 12. Four pictures demonstrating exposure problems. Taken from left: 1—a frame from evaluation 2 described in Section 4, 2—a sample frame from a webcam taken with automatic exposure settings, 3—a sample frame of the same subject under the same conditions as the second frame, but this time taken with exposure settings manually selected, 4—same picture as in 3, but showing a successful eye-pair detection.

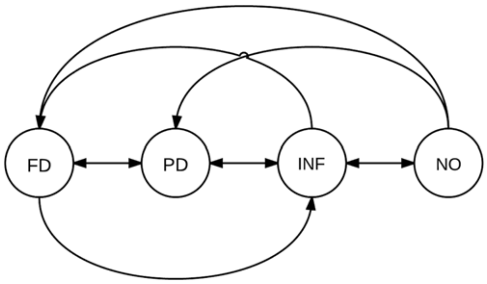


Fig. 13. A state flow diagram of the multi-level feedback system. FD stands for Full Detections, PD for Partial Detections, INF for Inference, and NO for No Data.

for computer vision algorithms to successfully detect the eyes. To verify this we tested both frames (with incorrect and correct exposure settings respectively) using our system. The rightmost picture in Fig. 12 shows that the system correctly identified the user's eyes when exposure settings were corrected (the eyes are indicated with a blue rectangle). However, the second picture from the left in Fig. 12 did not retain enough detail of the face for the system to detect the eyes. Interestingly, programmatic control of exposure settings is already part of a specification for USB video class devices. Unfortunately, many manufactures do not implement this specification, or they only implement a limited subset.

6.2. Recognition indicator

Our evaluations revealed that viewing distance cannot currently be reliably estimated in heterogeneous environments with varying lighting conditions. We therefore recommend implementing a recognition indicator that provides information to users if their eyes cannot be tracked by the system.

First it is important to note that our eye-distance estimation uses two levels of detection. In the first detection level the system identifies an eye-pair in the camera image as a rectangular block. This detection level is sufficient, but not ideal, for estimating relative eye-distance. Therefore we also use a second detection level that identifies the individual eyes within each subregion of the originally detected rectangular eye-pair region. If the eyes are detected at this level then we can use the pupils of the eyes as a basis for estimating relative eye-distance. This increases accuracy.

Fig. 13 shows a state flow diagram of our proposed indicator. It depicts a more advanced version of the system we described in the case study. With the exception of the inference state, all other states are already present and implemented in our system.

The first state is Full Detection (FD). It means that our algorithm was able to detect an eye-pair and confirm the presence of the eyes in each subregion. Therefore, the system is confident in both the correctness of detection of the eye-pair and in the accuracy of the distance estimation.

The second state is Partial Detection (PD). This is the first fallback stage. The algorithm has detected an eye-pair but it is unable to confirm the location of both eyes. Therefore, our confidence in the correctness of the detection of the eye-pair is lower. We can also no longer estimate the distance of the user from the display with high accuracy.

The third state is Inference (INF). The system enters this state when the vision algorithm fails to detect any eye-pairs in the camera stream. For a short amount of time (ca. <1 s) it is plausible we can infer a likely position of the user based on

previous data with reasonably high accuracy. We can also confirm that the user is still active based on, for example, touch and keyboard input. However, the reliability of our inference will decrease with time.

The last state is No Data (NO). The system enters this state when its confidence in being able to accurately infer the likely position of the user drops below a defined threshold.

In the above design, the amount of user feedback depends on our current detection state. In the Full Detection state the system is fully working. Since interaction with the system is expected to be subtle providing minimal or no feedback is appropriate.

In the Partial Detection state, the amount of feedback depends on how the system is intended to be used. If eye-distance is not used as a main interaction modality providing minimal to no feedback may be appropriate. However, if eye-distance is the primary interaction modality then it may be necessary to provide feedback to the user. If the state does not change within a few seconds, this is a sign of a calibration issue. After a set timeout within this state, the system can inform the user to adjust their device to try to improve detection. We recommend displaying a small icon depicting for example a crossed eye as an indication to the user. We suggest using a green colour as the system is still detecting the user.

In the Inference state the system needs to provide continuous feedback. For example, it can display a confidence bar that changes its size and colour depending on our confidence in our inference. This feedback can be coupled with the crossed eye icon we previously suggested for the Partial Detection state. This time, however, we suggest using an orange icon colour in order to convey that the system is not detecting the eyes at all. The system's confidence is decreasing as a function of the amount of time the system remains in this state.

In the No Data state the system cannot estimate eye-distances at all. We recommend using the same crossed eye icon as before. We suggest using a red colour to indicate that the system is longer able to estimate eye-distances at all.

7. Conclusions

In this paper we explored how we can use absolute and relative viewing distance between the user and the display as a complementary interaction modality. We first mapped out the design space for techniques using viewing distance. Thereafter we presented markerless techniques for reliably estimating absolute and relative viewing distance ($\geq$90% accuracy) for both mobile and desktop devices under constant lighting conditions using readily available computer vision algorithms and commodity cameras. We then investigated the accuracy of absolute and relative viewing distance estimation for our system in both indoor and outdoor environments. In our evaluations we found that viewing distance is a promising complementary input modality that can be reliably estimated using computer vision in environments with constant lightning. For environments with heterogeneous lightning conditions several challenges still exist when designing practical systems. To aid practitioners and researchers we therefore highlighted several design implications for future systems.

References

[1] A. Jaimes, N. Sebe, Multimodal human–computer interaction: a survey, Computer Vision and Image Understanding 108 (1–2) (2007) 116–134.
[2] E. Murphy-Chutorian, M.M. Trivedi, Head pose estimation in computer vision: a survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (4) (2009) 607–626.
[3] J. Yang, R. Stiefelbagen, U. Meier, A. Waibel, Visual tracking for multimodal human computer interaction, in: CHI'98 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 1998, pp. 140–147.
[4] M.-H. Yang, D.J. Kriegman, S. Member, N. Ahuja, Detecting faces in images: a survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (1) (2002) 34–58.
[5] S. Wang, X. Xiong, Y. Xu, C. Wang, W. Zhang, X. Dai, D. Zhang, Face-tracking as an augmented input in video games: enhancing presence, role-playing and control, in: CHI'06 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 2006, pp. 1097–1106.
[6] G.R. Bradski, Computer vision face tracking for use in a perceptual user interface, Intel Technology Journal (1998).
[7] L. Maat, M. Pantic, Gaze-X: adaptive, affective, multimodal interface for single-user office scenarios, Lecture Notes in Artificial Intelligence 4451 (2007) 251–271.
[8] R. Vertegaal, Designing attentive interfaces, in: ETRA'02 Proceedings of the 2002 Symposium on Eye Tracking Research & Applications, ACM Press, 2002.
[9] C. Dickie, R. Vertegaal, C. Sohn, D. Cheng, EyeLook: using attention to facilitate mobile media consumption, in: UIST'05 Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology, ACM, 2005, pp. 103–106.
[10] H. Drewes, A. De Luca, A. Schmidt, Eye-gaze interaction for mobile phones, in: Mobility'07 Proceedings of the 4th International Conference on Mobile Technology, Applications, and Systems and the 1st International Symposium on Computer Human Interaction in Mobile Technology, vol. 07, ACM, 2007, pp. 364–371.
[11] C. Harrison, A.K. Dey, Lean and zoom: proximity-aware user interface and content magnification, in: CHI'08 Proceedings of the Twenty-Sixth Annual SIGCHI Conference on Human Factors in Computing Systems, ACM, 2008, pp. 8–11.
[12] D. Vogel, R. Balakrishnan, Interactive public ambient displays: transitioning from implicit to explicit, public to personal, interaction with multiple users, in: UIST'04 Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology, ACM, 2004, pp. 137–146.
[13] T. Prante, C. Röcker, N. Streitz, R. Stenzel, C. Magerkurth, D. van Alphen, D. Plewe, Hello. Wall—beyond ambient displays, in: 5th International Conference on Ubiquitous Computing, Ubicomp'03, 2003.
[14] T. Ballendat, N. Marquardt, S. Greenberg, Proxemic interaction: designing for a proximity and orientation-aware environment, in: ITS'10 ACM International Conference on Interactive Tabletops and Surfaces, ACM, 2010, pp. 121–130.
[15] K. Khoshelham, Accuracy analysis of kinect depth data, Geo-Information Science 38 (5/W12) (2010).
[16] M.A. Nacenta, S. Sakurai, T. Yamaguchi, Y. Miki, Y. Itoh, Y. Kitamura, S. Subramanian, C. Gutwin, E-conic: a perspective-aware interface for multi-display environments, in: UIST'07 Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, ACM, New York, USA, 2007, pp. 279–288.
[17] M. Castrillón-Santana, O. Déniz-Suárez, L. Antón-Canalís, J. Navarro-Lorenzo, Face and facial feature detection evaluation: performance evaluation of public domain Haar detectors for face and facial feature detection, in: VISAPP 2008, 2008.
[18] P. Viola, M.J. Jones, Robust real-time face detection, International Journal of Computer Vision 57 (2) (2004) 137–154.