2017 14th Conference on Computer and Robot Vision

A Window to your Smartphone: Exploring Interaction and Communication in Immersive VR with Augmented Virtuality

Amit P. Desai
Department of Computer Science
Memorial University of Newfoundland
St. John's, Canada
*apd525*at*mun*dot*ca*

Lourdes Peña-Castillo
Department of Computer Science
Department of Biology
Memorial University of Newfoundland
St. John's, Canada
*lourdes*at*mun*dot*ca*

Oscar Meruvia-Pastor
Department of Computer Science
Office of the Dean of Science
Memorial University of Newfoundland
St. John's, Canada
*oscar*at*mun*dot*ca*

***Abstract*—A major drawback of most Head Mounted Displays (HMDs) used in immersive Virtual Reality (VR) is the visual and social isolation of users from their real-world surroundings while wearing these headsets. This partial isolation of users from the real-world might hinder social interactions with friends and family. To address this issue, we present a new method to allow people wearing VR HMDs to use their smartphones or tablets without removing their HMDs. To do this, we augment the scene inside the VR HMD with a view of the user's device so that the user can interact with the device without removing the headset. The idea involves the use of additional cameras, such as the Leap Motion device or a high-resolution RGB camera to capture the user's real-world surrounding and augment the virtual world with the content displayed on the smartphone screen. This setup allows VR users to have a window to their smartphone from within the virtual world and afford all of the functionality provided by their smartphones, with the potential to reduce some of the undesirable isolation users may experience when using immersive VR systems.**

***Keywords*-Augmented Virtuality; Virtual Reality; Social Isolation; Smartphone Detection; Head Mounted Displays; Smartphones; Logistic Regression; Linear Discriminant Analysis; Quadratic Discriminant Analysis;**

I. Introduction

Virtual Reality (VR) Head Mounted Displays (HMDs) have become increasingly popular among a wider segment of the population [1] thanks to the availability of low-cost hardware and sophisticated game engine software. A major part of the credit goes to products such as the Oculus Rift [2], HTC Vive [3], Samsung Gear VR [4], PlayStation VR [5], Google Cardboard [6] and similar technologies, which have extended the accessibility of VR to a wide segment of the population. These consumer devices provide a rich, interactive and immersive VR environment by offering low-cost, high-resolution stereoscopic images, wide field of view and, in some cases, high graphics quality. In addition, the use of Virtual Reality Head Mounted Displays (VR HMDs) is increasingly being applied for gaming, immersive and collaborative data visualization [7], training [8], and rehabilitation medicine [9]. Although VR HMDs are designed to provide an immersive environment where users feel they are part of the virtual environment, a major drawback of most immersive VR HMDs is that users cannot see and sometimes cannot hear their surrounding environment, partially becoming isolated from the real world [10]. Even small tasks like finding the keyboard or picking up a phone call can become difficult while wearing a VR HMD [11], and might lead to temporary isolation, where users are not able to get in touch with their friends and family when immersed in the VR space. One can argue that the mentioned difficulties can be avoided by having the users remove their HMDs, but such an interruption might have a negative effect on the VR experience or its outcomes, which might not be warranted, for instance, if a user just needs to check out a message or talk on the phone.

In the current world of social networks, users have a strong urge to be in touch with their friends and family through social media. According to the statistics published in [12], more than 2.3 billion people are active social media users. Among them, 1.9 billion users engage with social media using a mobile device. In the absence of inbuilt communication software inside the VR, each time a user needs to use the smartphone to get in touch with friends and family or check for messages, the user needs to remove the HMD which might deteriorate the VR experience. Through Mixed and Augmented Reality, additional devices such as an RGB camera attached to the HMD can be used as a window to the real world. However, finding the balance of Real and Virtual imagery that must be shown to the users is still hard, and it is might be difficult to read and perceive the text displayed on a smartphone or tablet's when it is captured through an RGB camera and rendered through a commercial grade VR HMD.

To address the temporary visual isolation of the users from the real world while wearing a VR HMD, we present a method that enables users to interact with their smartphone devices so that they can read messages, place calls, and use other apps while still being immersed in the VR environment. Users will no longer have to remove their HMD's

978-1-5386-2818-8/17 $31.00 © 2017 IEEE
DOI 10.1109/CRV.2017.16

each time they need to use their smartphone. To provide a window to the smartphone, our approach uses an infrared (IR) camera such as the Leap Motion device [13]. The images captured from the Leap Motion device are processed to detect the presence of the smartphone in the real-world. Once the smartphone is accurately detected in the real world, the scene in the VR world is augmented with the view of the user's smartphone, therefore enabling the users to use their smartphone without removing the headset.

The first step to augment the scene inside the VR HMD with a view of the user's device is to accurately detect the edges of the smartphone in real-time. A smartphone is a texture-less object, in the sense that it lacks any repeatable or constant image patterns on its screen and/or surface, that can be used by existing keypoint-based approaches to detect and track it. As keypoint-based approaches like SIFT [14], SURF [15] or ORB [16] could not be used, we decided to explore edge-based object detection to detect the smartphone in real-time and designed an algorithm using some basic image processing. The designed algorithm could successfully detect the smartphone in the Leap Motion images, but at the cost of a high number of incorrect detections (high false positive rate). We then decided to explore the use of machine learning approaches to improve the specificity of our system while maintaining a high recall rate. In sum, we first processed the images obtained from the Leap Motion device using standard image processing techniques for smartphone detection. Then, we identified and extracted several unique features (attributes) from the processed image, and gave these features to the statistical model to discriminate between correctly or incorrectly detected smartphone in the image. Once the smartphone is detected, the VR scene is augmented with the view of the user's smartphone based on the classifier's output.

The paper is organized as follows: Section II summarizes the previous work done in Augmented Virtuality, texture-less object detection and tracking, and previous instances of use of machine learning methods in image processing. Section III describes system setup, devices used and applications required to make the system work. Section IV describes the system design and implementation of image-based smartphone detection system (IBSD) and smartphone detection system using statistical classifiers (SDSC). Section V discusses our achievement, presents results and describes limitations of our system. Section VI lists future enhancements and summarizes our findings.

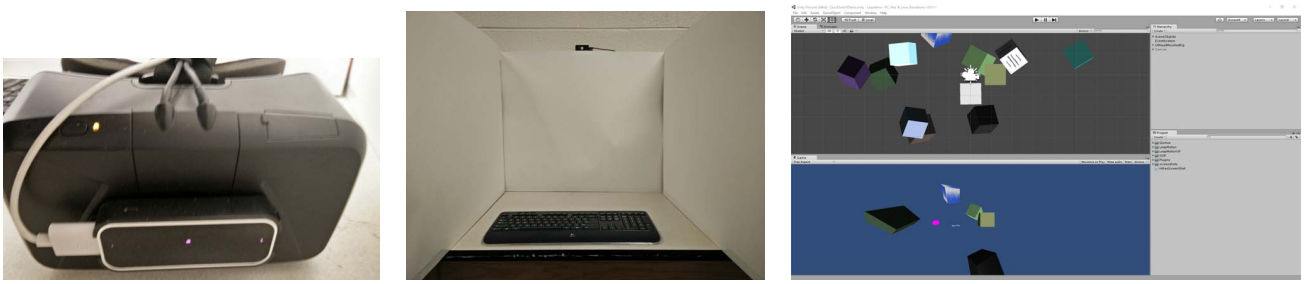
II. Background

A. Never Blind in VR: Augmented Virtuality to the rescue

While wearing an HMD, interacting with real world objects is still a challenge, as users are partially isolated from the outside environment. They cannot perceive most of it because their sight is occluded by the HMD and sometimes they cannot even hear their surrounding environment if they are wearing headphones inside the VR. Under these conditions, interacting with other people and finding objects like desktops and peripherals or a cup of coffee is difficult [11]. To address this, solutions usually involve the use of 'Augmented Virtuality', where real world objects are embedded in the virtual environment. These solutions embed user's hands and/or objects that are in close proximity of user's interaction space with the help of a head-mounted RGB camera, a head-mounted RGBD camera, or external depth sensing devices like Microsofts Kinect.

Among the early work that involved combination of real and virtual world elements in a scene was the SIMNET project [17]. The system developed there could overlay real-world images onto a virtual world scene. With the help of a head-mounted camera placed on an HMD, the proposed system was able to provide a window to the outside world from within the virtual environment. This enabled the users to perceive real world objects within the virtual world. A more recent solution that involved the use of 'Augmented Virtuality' was presented in 2009 by Steinicke et al [18]. The solution presented was able to model a user's hands and body inside the virtual environment. It was achieved with the help of a head-mounted RGB camera and chroma-keying technique. In 2014, [19] demonstrated a system that allows the users to see and use their hands while interacting with virtual objects placed around the virtual environment. The demonstrated system consists of an RGBD camera that captures users hands in real-time and embeds them in the virtual world. While in 2015, [11] presented a user study to help identify challenges faced by the users of HMD. The study used an external head-mounted camera to embed peripheral devices like the keyboard in the virtual environment to examine users ability to interact with real world objects.

Other researchers have been able to address other aspects of the isolation issue using Microsoft's Kinect, a depth sensing camera which can provide visual feedback of the objects that surround the user. The Kinect allows room-wide sensing capabilities, which makes it ideal for user tracking and gesture identification. One such system was presented by Dassault Systémes [20]. The system presented was successfully able to embed 3D point clouds obtained by the Kinect V2 in the user's virtual space and display elements of the real world that are in close proximity to the user. Although the system was not portable, they were able to display point clouds that represent people present in the same room, allowing for people to see each other in the virtual world and throw and catch virtual objects at each other, partially addressing the social isolation issue. Another research conducted by [11] that involved a study of engagement dependent awareness of other people present in the same room was presented in 2015. Similar to the system presented by 'Dassault Systémes', the authors used the Kinect to insert other people's silhouettes whenever they



(a) (b) (c)

Figure 1. System Setup: (a) Leap Motion device attached to Oculus Rift (b) Cubicle to test IBSD (c) VR application created in Unity

are within a close proximity to the user. In order to make it engagement-dependent, the other people's outline is initially faded in the VR world and then if the user wishes to engage, the users became visible in full color.

So far, most of the research done in this area has focused on to make the user aware of other users present in close proximity, and little research has been done to help the user stay in touch with the part of the world which is not in close physical proximity. The main motivation behind this research is to allow people to stay in touch with others and their digital life by allowing them to use their personal computing and communication devices, such as smartphones and tablets.

B. Texture-less object detection and tracking

Existing keypoint-based object detection and tracking approaches like SIFT [14], SURF [15] or ORB [16], rely on matching descriptor features like blobs, corners or patterns in texture, but may not be effective in detecting featureless objects [21], [22]. As a smartphone device is a texture-less object, in the sense that it lacks any repeatable and constant image patterns on its screen and/or surface, that can be used to detect and track it using existing keypoint-based techniques. As keypoint-based approaches like SIFT, SURF or ORB could not be used, texture-less object detection techniques were explored. Most of the state-of-the-art techniques present in the area of texture-less detection use edge-based template matching [21], [23]–[26]. Although, these techniques perform better than keypoint-based technique for texture-less object detection, they suffer from problems related to occlusion, cluttering and scalability of template library [22]. Also, there have been many instances where efforts have been made to improve the search time required for template based matching [23]–[25], while some other researchers have been able to perform this in real-time [21], [27]–[29]. In our case, as smartphone's vary in different sizes and aspect ratio, existing template-matching texture-less object detection methods may require a large template database. Ideally the template database would consist of every possible 3D position of the smartphone as well as each variation of aspect ratio, making it inefficient for real-time tracking. In order to avoid an initial large template database and real-time smartphone tracking, we decided to make use of traditional edge based object detection method as described in Section IV.

C. Machine learning methods in image processing and HCI

There are many instances of the use of machine learning methods in image processing and human-computer interaction (HCI). Artificial Neural Networks (ANNs) and K-Nearest Neighbors (K-NNs) were used to classify facial expression to help in assessing user's feelings, characteristics, and intentions [30]. In this work, Hai et al. were able to achieve a classification precision of 92.38%. [31] made use of ANNs and Support Vector Machines (SVMs) to develop a hand recognition system. The system designed was able to recognize seven different gestures with 99% recognition accuracy. [32] used SVMs to automatically recognize human postures with an accuracy of 99.14%, while [33] used features extracted from images to train classifiers to categorize personal photos into privacy classes.

Machine learning is also extensively used in medical applications that involve processing of images captured from various medical instruments to identify a disease. For example, in [34], Moreira et al used LDA and SVM classification models to identify and evaluate Lymphedema impairments in Breast Cancer Patients. The researchers used a Kinect to capture and extract features from the upper-limbs motion of the body to identify lymphedema, which is caused commonly by the removal or damage of the lymph nodes in cancer patients, with a misclassification error of 19%. In another instance, [35] applied SVMs to capture and extract features from magnetic resonance images (MRIs) to identify patients suffering from Schizophrenia with 83.33% accuracy.

III. Setup

Our approach uses an additional camera such as the Leap Motion [13], to provide a window to the real world inside VR application. Leap Motion is attached to Oculus Rift DK2 device as shown in the Figure 1(a) using a Universal VR Dev Mount [36]. The smartphone device used was a One Plus One with 3 GB of RAM, Qualcomm Snapdragon 801

Quad-core processor, 5.5 inches of display and 1080 x 1920 pixels (401 PPI pixel density) screen resolution. We also designed an Android App that will transfer the screenshots and orientation data captured from the smartphone device to the VR application. The Android App is designed to act as a TCP/IP server and transfer screenshots as well as smartphone's orientation data to the VR application. The VR application was designed using the Unity game engine. Cubes of different colors were placed at random locations in the VR space and a Unity Quad object, which changed shape according to the detected smartphone, was used to display the smartphone's screenshots. The images captured from the Leap Motion device were displayed in the background of the VR space such that it would occupy the entire VR space giving a perspective of a see-through window to the real-world from the VR environment. The VR application also acts as a TCP/IP client and connects to the Android app via local LAN to receive screenshots and orientation data. Figure 1(c) shows the designed VR application. In addition, to test the smartphone detection algorithm, we set up a cubicle as shown in Figure 1(b) to avoid limited interference from other objects surrounding the user [11].

IV. System Design and Implementation

A. Image-Based Smartphone Detector (IBSD)

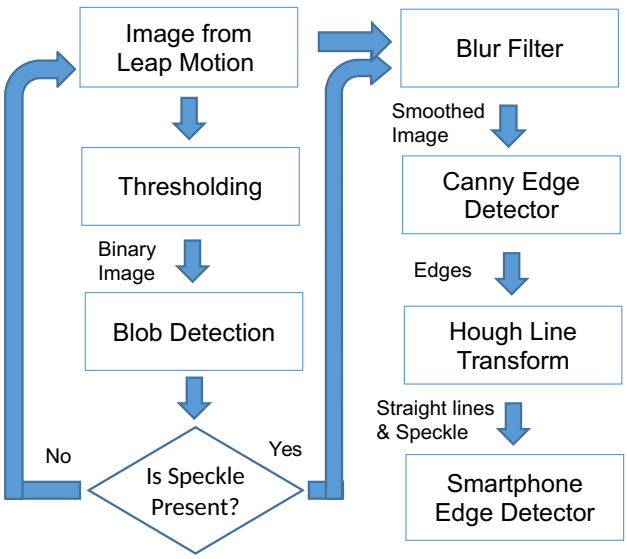


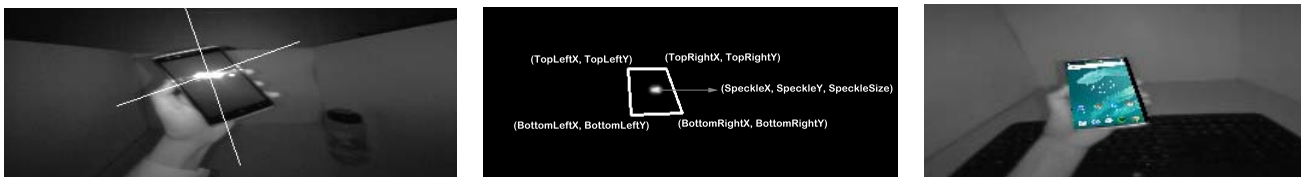
Figure 2. Flow diagram of the smartphone detector

Steps involved in detecting a smartphone in an image captured by the Leap Motion device are outlined in Figure 2. All the image processing steps were performed using OpenCV. An OpenCV wrapper known as EmguCV was used to access OpenCV API's in C# language. Detection of any object in an image involves two steps: locate the region of interest and then recognize if the desired object is present in that region or not. In our case, instead of searching the entire image for the smartphone, we relied on a unique finding that helped us localize the smartphone: The Leap Motion device is basically an Infrared (IR) Camera that relies on the reflected IR light back to its sensors to detect user's hands. We noticed that whenever an object which has a reflective surface such as the smartphone is held in front of the Leap Motion device, it produces a blob with high brightness (which we call smartphone's speckle) as seen in Figure 3(a). We used this unique property to localize the smartphone in the Leap Motion image. Also, our IBSD algorithm assumes that users will position their smartphone in front of the HMD in such a way that they can read and/or view the smartphone's screen correctly, i.e., the smartphone's screen is facing the HMD.

The smartphone detection algorithm works as follows: the VR application first captures grayscale images from the Leap Motion device through a set of API's provided by Leap Motion. We apply a fixed-level threshold (binary threshold where T=230) on the grayscale image and with the help of a blob detection algorithm find the smartphone's speckle. Once the smartphone is localized, we extract a region of interest (ROI) (radius of 75 pixels) around the speckle from the original grayscale image obtained from leap motion. We apply a normalized box filter (with kernel size = 3x3) to smooth the extracted image and detect edges with help of the Canny edge detector. Smartphones have a particular geometry, i.e., they are rectangular and have straight edges. We use a basic principle in perspective projection for detecting these straight smartphone's edges, i.e., in perspective projection straight lines in 3D will project to straight lines in the 2D image. We apply a Probabilistic Hough Line Transform algorithm available in OpenCV to detect these straight lines. Once the straight lines are detected, we pass these lines (potential smartphone edges) through the smartphone edge detector module. The smartphone edge detector module then applies some simple mathematical functions on the extracted ROI image to detect if a potential smartphone edge belongs to the smartphone or not. With the smartphone's speckle as the center, the smartphone detector module shoots four straight lines in four different directions (Edge Detector Lines) with respect to the current orientation of the smartphone, as shown in Figure 3(a), and extracts only those potential smartphone edges that intersect with these detector lines. The angles of extracted edges are then compared and only those pairs that are nearly parallel to each other are considered as smartphone edges.

B. Smartphone Detection with Statistical Classifier (SDSC)

The IBSD was able to detect the smartphone's edges; however, there were many instances where edges of other objects in the surroundings were also considered a smartphone edge. In short, our devised algorithm had a high false positive rate, which was unacceptable. Figure 5(b) shows



(a) (b) (c)

Figure 3. (a) Image captured by leap motion device with speckle at the center of the cross hairs (b) Features extracted from the detected smartphone (c) A sample instance where SCSD was able to detect smartphone edges

instances where edges of other objects in the surroundings were considered to be edges of the smartphone. A detected false smartphone edge will not only produce undesirable results while augmenting the smartphone device in VR environment but might also place the user's fingers in a false position in the VR scene, making it difficult for the users to use their smartphone. To lower the false positive rate and obtain better detection results, we decided to explore the use of machine learning approaches, such as statistical classifiers, that would be able to determine whether or not the detected edges are indeed the smartphone's edges. We required a classifier that could satisfy constraints such as detection of smartphone in real-time, ease of integration with the existing solution, possibly by some mathematical model, no additional process calls to external programs or libraries that might slow down the smartphone detector, and no additional storage of instance vectors (as required by instance-based classifiers such as K-NN). Considering the constraints, we decided to evaluate three statistical classifiers: Logistic Regression (LR) [37], Linear Discriminant Analysis (LDA) [37], and Quadratic Discriminant Analysis (QDA) [37]. These classifiers generate mathematical models that can be easily integrated into the existing system and do not cause significant computing overhead. To select the most suitable classifier we assessed the predictive performance of these three statistical learning methods and selected the one with the lowest false positive rate at a comparable recall rate. Once the images captured by the Leap Motion device are processed, we extract a set of 11 features from the detected edges. Figure 3(b) illustrates these features in the processed images. Since the values of the extracted features are absolute X and Y coordinates of the smartphone detected in the image, these features are normalized.

To implement smartphone detection with a statistical classifier, we first generated a data set and evaluated the performance of three classifiers (LR, LDA, and QDA). The dataset contained 312 instances or records. Each instance was made up of a set of 11 features (see Figure 3(b)) that were extracted from the smartphone's edges detected by IBSD as described in the previous section. The instances were classified into two groups depending on whether the edges of the smartphone are correctly detected (group D) or not (group N). For each instance, a feature vector, along with its corresponding class, is stored in the dataset. Out of 312 records, 222 (or 71.15%) records belonged to group N while 90 (or 28.85%) records belonged to group D. To estimate the performance measurements for each statistical learner (LR, LDA, and QDA), we carried out 10-fold cross-validation. At the end of the 10-fold cross-validation, we obtained the average and standard deviation of each performance measurement. Table I gives the performance measures averaged out over the ten folds for each classifier. The classifier with the lowest false positive rate (< 0.2) at a comparable recall of 0.8 was QDA (Figure 4), and thus QDA was selected to be used in the SDSC implementation. QDA was then trained with the complete dataset to generate the classification model to be implemented. To do this, we obtained class means per feature and class covariances from the dataset. These values were then used to generate a mathematical model that was integrated with our existing smartphone detector. For QDA, the Bayes classifier can be approximated by the equation [37]:

Table I
AVERAGE PERFORMANCE MEASURES AUC ($\pm$ SD), SENSITIVITY ($\pm$ SD), SPECIFICITY ($\pm$ SD) AND ACCURACY ($\pm$ SD) OF LR, LDA AND QDA OVER 10 FOLDS. SENSITIVITY, SPECIFICITY AND ACCURACY OF LR, LDA AND QDA FOR EACH OF THE 10 FOLDS WERE CALCULATED FOR THE OPTIMAL PROBABILITY CUTOFF. THE AVERAGE OPTIMAL CUTOFF FOR LR, LDA AND QDA WAS 0.295 ± 0.10, 0.280 ± 0.10 AND 0.642 ± 0.30 RESPECTIVELY

Performance Measure	**LR**	**LDA**	**QDA**
AUC	0.778 ± 0.13	0.777 ± 0.14	0.935 ± 0.04
Sensitivity	0.875 ± 0.12	0.873 ± 0.09	0.906 ± 0.09
Specificity	0.710 ± 0.17	0.701 ± 0.17	0.899 ± 0.07
Accuracy	0.758 ± 0.13	0.754 ± 0.12	0.907 ± 0.06

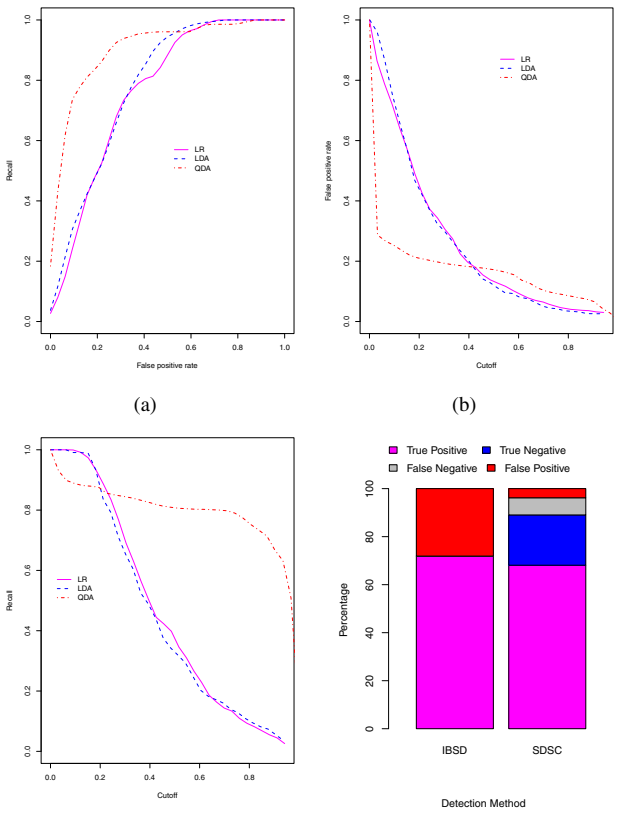
$$\delta_k(x) = -\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) + log\pi_k \qquad (1)$$

where μ_k = mean of k$^{\text{th}}$ class, Σ_k = covariance matrix for the k$^{\text{th}}$ class, π_k is prior probability of k^{th} class and $\delta_k(x)$ is the discriminant function for the class k. Observation x is assigned to the class for which $\delta_k(x)$ is largest.

V. RESULTS AND DISCUSSION

A. IBSD

By using the smartphone speckle and traditional image processing algorithms, we were successfully able to quickly



(a) (b) (c) (d)

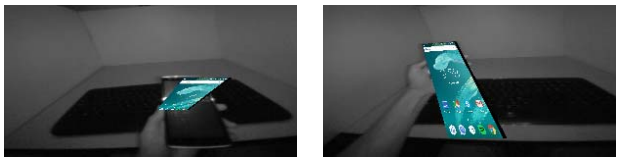
Figure 4. (a) Comparison of LR, LDA and QDA based on ROC curve across various probability thresholds (b) Comparison of LR, LDA and QDA based on recall (sensitivity) across various probability thresholds (c) Comparison of LR, LDA and QDA based on false positive rate (1 - specificity) across various probability thresholds (d) Comparison between IBSD and SDSC. The y-axis indicates the percentage of instances in the validation set. IBSD error rate is 28.1% (red block) while SDSC error rate is 11% (gray and red blocks).

localize and recognize a smartphone in a given leap motion image. Although there were certain instances where edges from surrounding objects were incorrectly identified as smartphone edges, IBSD was able to successfully detect a smartphone nearly 70% of the time. There are certain situations where IBSD typically fails, one such instance is when the smartphone's screen is facing HMD in such an angle that the speckle will not be present on the smartphone. However, this would imply that users are holding the smartphone in such a way that they will not usually be able to read clearly the text displayed on the smartphone's screen. Another instance where IBSD fails is when a similar object with a highly reflective surface is present in the near vicinity of the smartphone. In these cases, there will be two speckles present in the leap motion image, and IBSD may produce a false detection when the wrong speckle is chosen.

B. Comparison of IBSD against SDSC

After integrating the mathematical model of the statistical classifier into IBSD, we further compared the performance of IBSD against that of SDSC with a validation data set that was not used during training. To do that, a dataset of 210 records was generated. Out of 210 records obtained, 59 (or 28.1%) records belonged to group N while 151 (or 71.9%) records belonged to group D. Out of the 210 instances, there were 59 (or 28.1%) instances where IBSD considered edges of surrounding objects to be part of the smartphone, while only 8 (or 3.8%) false positives were reported by SDSC (Figure 4(d)). SDSCs total error rate was 11%.

As higher specificity means that the devised algorithm rejects most of the false positive cases (improper detection of smartphone edges), we were mostly interested in reducing the false positive rate. Comparing IBSD and SDSC, we found that SDSC rejects 75% of the cases that were falsely considered to be smartphone by IBSD. An accuracy of 89% was achieved by SDSC compared to an accuracy of 72% obtained by IBSD. Some instances from the validation dataset where the IBSD incorrectly detects the edges of a smartphone are shown in Figure 5. Thus, with the help of statistic classifiers, we were able to improve the accuracy when deciding which detected edges truly correspond to the edges of the smartphone. SDSC works as an additional filter which discards false smartphone edges detected by IBSD. Therefore, it will still be limited by the inability to detect a smartphone when the speckle is missing, but will perform better at discarding false positives when the wrong speckle is identified.



(a) (b)

Figure 5. Some sample instances that represent incorrect smartphone detection by IBSD. Note that in these images the edges of other objects in the surroundings are considered part of the smartphone. SDSC successfully rejects these incorrect edge labelling.

C. Smartphone detection after SDSC

By combining traditional image processing techniques with statistical classifiers, we designed a system that can successfully detect the presence of the smartphone in 90% of the true positive cases. Our empiric assessment indicates that the system can respond at interactive rates. The system is also able to map the images obtained from smartphone's screen to the smartphone displayed inside the VR space, and our initial testing showed that user can operate their smartphone device without having to remove their HMD device. Some instances where the user was able to launch an App and/or able to read text displayed on the smartphone's screen in VR environment are shown in Figure 6. These images show that, although limitations exist, the method proposed can bring modern communication devices like

smartphone inside the VR space. It shows that users can operate their smartphone from inside the VR space and no longer have to remove their HMD's each and every time they need to check their social networks or talk with friends and family over the phone.

The main limitation we currently observe with the produced system is the jumpiness of the smartphone display within the virtual environment. This jumpiness is due to several factors, such as the low resolution of the images provided by the Leap Motion device, but mostly arises from the instability of the algorithms in the image-processing pipeline, which are sensitive to noise from the users fingers. In addition, the high resolution of the smartphone is filtered through the low resolution of the HMD, while the natural motion of the users head relative to the users hand while holding the phone guarantees that the input will always be in flux.

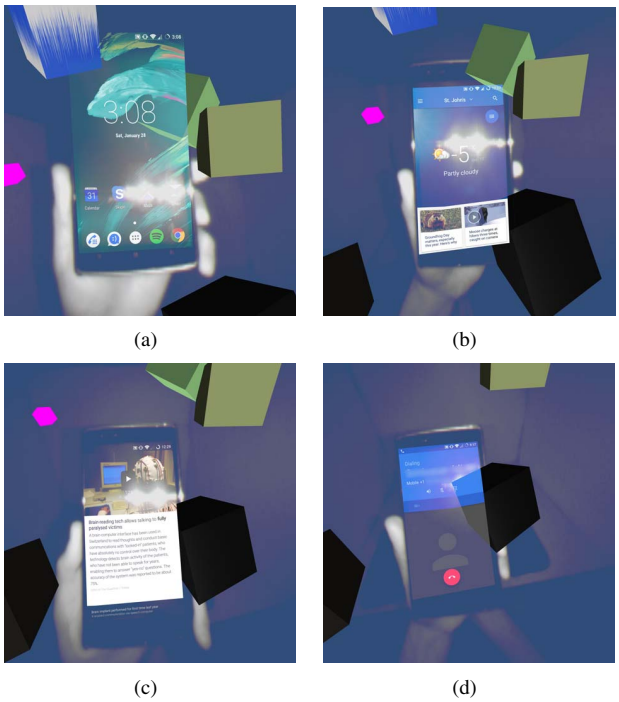


Figure 6. Some instances where users were able to interact with the smartphone inside VR environment. (a) Smartphone's home screen mapped to the smartphone displayed in Leap Motion Image (b) Users can check weather updates (c) Users can read news in 'InShorts' app (d) Users can initiate a phone call

VI. Conclusions and Future Work

The visual isolation from the real world while wearing an HMD makes the task of using a smartphone difficult. This may in turn lead to temporary isolation, where users are not able to get in touch with their friends and family or check out important notifications when immersed in the VR space. To address this issue, we present a novel method that allows people wearing VR HMDs to use their smartphones without removing their HMDs. The devised method combines traditional image processing techniques with a statistical classifier to accurately detect the presence of the smartphone in the user's real-world surrounding. An additional camera such as the Leap Motion device is used to capture the user's real-world immediate surroundings. Once the smartphone is detected, we augment the scene inside the VR HMD with a view of the user's device so that the user can interact with the device without removing the headset. This setup allows the users to interact with the smartphone or tablet device from within the virtual space and be in touch with their friends and family through it. Although the proposed system has some limitations and the human visual system is quite sensitive to these, achieving a satisfying solution is within reach, and the presented results are encouraging. The dataset used to train the machine learning model is limited. Smartphones of different sizes or variations in the way users hold their smartphones at different positions and orientations are not considered. The dataset will be further enriched to include data generated with various smartphones, user postures and orientations. The proposed system will be further explored for use in application areas where the use of traditional input devices can be challenging and smartphones may come in handy. Through user studies, activities that rely on the smartphones touch screen as an input device, such as text entry, game control, internet browsing, and drawing gestures will be further explored.

Acknowledgment

The authors would like to thank NSERC for funding through a Discovery Grant and Dr. Ed Brown for supporting this project.

References

[1] D. Kushner, "Virtual reality's moment," *IEEE Spectrum*, vol. 51, no. 1, pp. 34–37, January 2014.

[2] Oculus, "Oculus Rift," https://www.oculus.com/en-us/rift/, 2016, accessed: 2016-05-01.

[3] HTC, "HTC Vive," http://www.htcvive.com/, 2016, accessed: 2016-05-01.

[4] Samsung Electronics Co. Ltd., "Samsung Gear VR," http://www.samsung.com/global/galaxy/gear-vr/, 2016, accessed: 2017-01-21.

[5] Sony Interactive Entertainment, "PlayStation VR," https://www.playstation.com/en-us/explore/playstation-vr/, 2016, accessed: 2016-05-01.

[6] Google VR, "Google Cardboard," https://vr.google.com/cardboard/index.html, 2016, accessed: 2016-05-01.

[7] C. Donalek, S. G. Djorgovski, A. Cioc, A. Wang, J. Zhang, E. Lawler, S. Yeh, A. Mahabal, M. Graham, A. Drake, S. Davidoff, J. S. Norris, and G. Longo, "Immersive and collaborative data visualization using virtual reality platforms," in *Big Data (Big Data), 2014 IEEE International Conference on*, Oct 2014, pp. 609–614.

[8] A. S. Mathur, "Low cost virtual reality for medical training," in *2015 IEEE Virtual Reality (VR)*, March 2015, pp. 345–346.

[9] L. K. Simone, M. T. Schultheis, J. Rebimbas, and S. R. Millis, "Head-mounted displays for clinical virtual reality applications: pitfalls in understanding user behavior while using technology," *Cyberpsychol Behav*, vol. 9, no. 5, pp. 591–602, Oct 2006.

[10] M. Billinghurst and H. Kato, "Collaborative mixed reality," in *Proceedings of the International Symposium on Mixed Reality*, Yokohama, Japan, March 1999, pp. 261–284.

[11] M. McGill, D. Boland, R. Murray-Smith, and S. Brewster, "A dose of reality: Overcoming usability challenges in VR head-mounted displays," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, ser. CHI 2015. New York, NY, USA: ACM, 2015, pp. 2143–2152. [Online]. Available: http://doi.acm.org/10.1145/2702123.2702382

[12] Smartinsights, "Social Media Stats," http://www.smartinsights.com/social-media-marketing/social-media-strategy/new-global-social-media-research/, accessed: 2016-05-01.

[13] Leap Motion, "Leap Motion VR," https://www.leapmotion.com/product/vr, 2016, accessed: 2016-05-01.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94

[15] H. Bay, T. Tuytelaars, and L. Van Gool, *SURF: Speeded Up Robust Features*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 404–417.

[16] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 2564–2571. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2011.6126544

[17] P. J. Metzger, "Adding reality to the virtual," in *Virtual Reality Annual International Symposium, 1993., 1993 IEEE*, Sep 1993, pp. 7–13.

[18] F. Steinicke, G. Bruder, K. Rothaus, and K. Hinrichs, "Poster: A virtual body for augmented virtuality by chroma-keying of egocentric videos," in *3D User Interfaces, 2009. 3DUI 2009. IEEE Symposium on*, March 2009, pp. 125–126.

[19] F. Tecchia, G. Avveduto, R. Brondi, M. Carrozzino, M. Bergamasco, and L. Alem, "I'm in vr!: Using your own hands in a fully immersive mr system," in *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, ser. VRST '14. New York, NY, USA: ACM, 2014, pp. 73–76. [Online]. Available: http://doi.acm.org/10.1145/2671015.2671123

[20] D. Nahon, G. Subileau, and B. Capel, "Never blind vr - enhancing the virtual reality headset experience with augmented virtuality," in *2015 IEEE Virtual Reality (VR)*, March 2015, pp. 347–348.

[21] C. Choi and H. I. Christensen, "3d textureless object detection and tracking: An edge-based approach," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 3877–3884.

[22] F. Tombari, A. Franchi, and L. Di, "Bold features to detect texture-less objects," in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 1265–1272.

[23] S. Hinterstoisser, V. Lepetit, S. Ilic, P. Fua, and N. Navab, "Dominant orientation templates for real-time detection of texture-less objects," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 2257–2264.

[24] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit, "Gradient response maps for real-time detection of textureless objects," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 876–888, May 2012.

[25] C. Steger, "Occlusion, clutter, and illumination invariant object recognition," *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, vol. 34, no. 3/A, pp. 345–350, 2002.

[26] M. Ulrich, C. Wiedemann, and C. Steger, "Combining scale-space and similarity-based aspect graphs for fast 3d object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1902–1914, Oct 2012.

[27] L. Vacchetti, V. Lepetit, and P. Fua, "Combining edge and texture information for real-time accurate 3d camera tracking," in *Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, Nov 2004, pp. 48–56.

[28] C. Kemp and T. Drummond, "Dynamic measurement clustering to aid real time tracking," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2, Oct 2005, pp. 1500–1507 Vol. 2.

[29] D. Schreiber, C. Beleznai, and M. Rauter, "Gpu-accelerated human detection using fast directional chamfer matching," in *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, June 2013, pp. 614–621.

[30] T. S. Hai, L. H. Thai, and N. T. Thuy, "Facial expression classification using artificial neural network and k-nearest neighbor," *International Journal of Information Technology and Computer Science (IJITCS)*, vol. 7, no. 3, p. 27, February 2015.

[31] G. Marqués and K. Basterretxea, "Efficient algorithms for accelerometer-based wearable hand gesture recognition systems," in *13th IEEE International Conference on Embedded and Ubiquitous Computing (EUC), 2015*, Oct 2015, pp. 132–139.

[32] Z. Zhang, Y. Liu, A. Li, and M. Wang, "A novel method for user-defined human posture recognition using kinect," in *7th International Congress on Image and Signal Processing (CISP), 2014*. IEEE, Oct 2014, pp. 36–740.

[33] D. Buschek, M. Bader, E. von Zezschwitz, and A. De Luca, *Automatic Privacy Classification of Personal Photos*. Cham: Springer International Publishing, 2015, pp. 428–435.

[34] R. Moreira, A. Magalhães, and H. P. Oliveira, *A Kinect-Based System to Assess Lymphedema Impairments in Breast Cancer Patients*. Cham: Springer International Publishing, 2015, pp. 228–236.

[35] E. Veronese, U. Castellani, D. Peruzzo, M. Bellani, and P. Brambilla, "Machine learning approaches: from theory to application in schizophrenia," *Computational and mathematical methods in medicine*, vol. 2013, p. 12, 2013. [Online]. Available: http://dx.doi.org/10.1155/2013/867924

[36] Leap Motion Inc., "Universal VR dev mount," http://store-us.leapmotion.com/products/universal-vr-mount-pre-order/, 2016, accessed: 2017-01-21.

[37] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning with applications in R*. Springer, 2013, vol. 6.