IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 30, 2020, accepted March 1, 2020, date of publication March 6, 2020, date of current version March 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2978899

3D Face Authentication Software Test Automation

DEBDEEP BANERJEE, (Member, IEEE), AND KEVIN YU
Qualcomm Technologies, Inc., San Diego, CA 92121, USA
Corresponding author: Debdeep Banerjee (debdeep.banerjee@gmail.com)

ABSTRACT The 3D face authentication has become a hot trend for researchers and developers in the recent years, due to its many advantages over the 2D face recognition feature. The 3D reconstruction of a human face using both near-infrared and depth sensors is a complex process on mobile phones. It commonly involves algorithms like face detection, face landmark detection, facial feature extraction, and depth information analysis. The 3D face authentication feature is critical for the user as per as security and also providing a convenient way to authenticate by the correct user. Therefore, the testing of 3D face authentication algorithms and applications in terms of functionality, performance, and stability is critical. However, the research on 3D face authentication application level validation and testing method is lacking in the field. Most testers are still validating the application manually. In this paper, we propose a robotic-arm-based test automation for testing the 3D face authentication feature on mobile phones. We programmed a 6 degree of freedom (6-DOF) robotic arm to perform 3D face authentication automated tests that were executed manually before. Our test automation also benchmarked the performance of an in-house developed 3D face authentication application and a 3rd party application which yielded promising latency and accuracy comparison results under different performance-impacting test scenarios.

INDEX TERMS Automation, robotics, software testing, software engineering.

I. INTRODUCTION

One of the main objectives of a software validation team is to effectively test the performance of the software. In this paper, we will focus on the design of a reliable test automation that can thoroughly evaluate the performance of a 3D face authentication end-to-end application. Creating a 3D Face authentication application on a mobile phone is challenging because multiple computer vision algorithms and security algorithms are utilized to ensure that the correct user is authenticated. Fig. 1 shows a near-infrared (NIR) image and a depth image that are provided by the 3D face authentication application.

We have developed a test automation that ensures that 3D face enrollment and authentication are automated. This test automation has helped us increase the testing efficiency. The main contributions of this paper are as follows:

1. An end-to-end robotic-arm-based test automation is developed for the execution of mobile device 3D face authentication tests under various angular movements, speeds and distances from the test subjects.

The associate editor coordinating the review of this manuscript and approving it for publication was Ehsan Asadi.

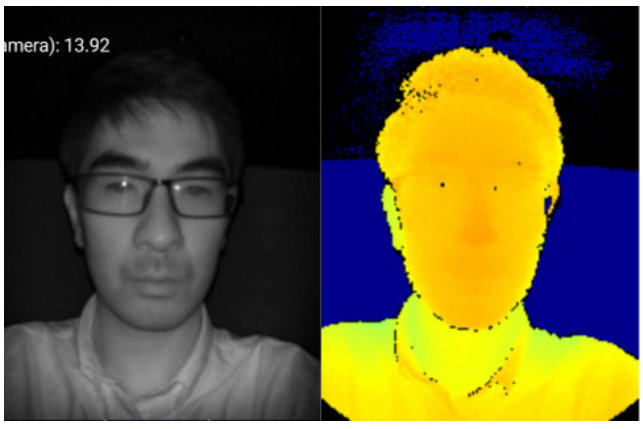


FIGURE 1. Near-infrared (NIR) image and depth image from the 3D face authentication device sensors.

The test automation realizes end-to-end automation of a 3D face authentication application by launching the application, focusing on the 3D face, and performing face authentication to determine whether to allow the user to access the phone software. The robotic arm is programmed to pick up the mobile phone being tested, focus the phone on the test subject, perform tests under various distances and angular movements, and perform 3D scanning of the test subjects

under various speeds to validate the 3D face authentication algorithm.

2. ***A test automation with live people is developed for conducting gaze detection tests.***

The robotic arm is programmed to perform 3D face authentication on real people using mobile phones. The robot is programmed to obtain 3D scans of the faces of people who are sitting in a designated location. The robot scans each subject's face and ensures that the irises were scanned so that gaze and liveness detection tests can be performed. This test can demonstrate whether the face authentication algorithm can authenticate real humans.

3. ***A false-rejection rate (FRR) test automation for 3D face authentication tests is proposed.***

If two users have enrolled their faces in the 3D face authentication algorithm application, then they should always be authenticated successfully. The rate at which users who have been successfully enrolled are not authenticated is the false-rejection rate. The objective of the algorithm is to realize a false-rejection rate of zero as we do not want to fail to authenticate users who have successfully enrolled their faces in the 3D face authentication application on a mobile phone.

4. The authentication efficiency under real-life test conditions, such as blurriness due to motion, is evaluated.

Real-world test scenarios must be automated to ensure that we can simulate the failures in field testing prior to testing in the functional test suite. The robotic arm is programmed to perform test cases with jitter while obtaining a 3D scan of a face; in this way, we simulate injection jitter scenarios while performing the 3D face authentication tests. Jitter can easily occur when a real user uses the application to authenticate himself/herself in a grocery store, e.g., to pay bills using mobile payment gateways.

II. MOTIVATION

The main objectives in developing a 3D face authentication test automation that uses a robotic arm are as follows:

1. Design and develop a reliable test automation for thoroughly testing the 3D face authentication feature via functional and performance tests.

Testing face authentication using 3D face recognition and authentication is tedious and error-prone. Using a programmable robotic arm, we have constructed a test setup that can be used to test 3D face authentication algorithms in a repeatable manner.

2. Test real-life scenarios with blurred vision during device motion scenarios.

The main objective of product testing is to ensure the requisite test automation capabilities for simulating real-world users and identifying issues early. Resolving these issues, if possible, will facilitate expedited product commercialization efforts.

3. ***Test the 3D face authentication feature under various angular motions and distances from the test subject. The robotic arm performs these precise movements.***

The 3D face authentication tests required repeated 3D scanning of a face under various speeds, angular movements, and distances. These tests were conducted to provide good test coverage of faces with various angles and computer vision algorithms were utilized to thoroughly test the face authentication features under these test scenarios.

4. ***Execute the test automation with real people.***

We needed a test setup in which we could perform face enrollment and authentication tests. Therefore, an end-to-end on-device test automaton setup was needed for automatically picking up the mobile phone under test, 3D scanning the users, and performing tests for evaluating the authentication algorithms.

III. BACKGROUND AND RELATED WORK

Robots have been used for the performance evaluation of path planning and learning algorithms [1]. To test unmanned vehicle systems for autonomous cars testbeds, such as real-time indoor autonomous vehicles, various test environments have been used [2]. Data aggregation for multisensory fusion, tracking, and localization is challenging for robots. Research has been conducted on developing tests for thoroughly evaluating robot movement solutions [3]. Data from multisensors can be integrated to optimize the use of robot movements [4]. Simulations that are based on software libraries and tools are needed for 3D visualization of robot trajectory planning and control [5]. For robots to visualize 3D space libraries, a point cloud library is needed for 3D generation [6], [7].

For the generation of a 3D point cloud, we must extract an optimal set that best characterizes a specified point cloud [8]. The robot operating system (ROS) is widely used to simulate robotic control and movements prior to deploying them on active assembly lines to perform tasks [9]. ROS code can be generated from models that foster fast software development for simulating robot control and movements [10].

Software testing has been accomplished by the programming of robots [11]. Robots can be programmed to perform repeatable tasks; this programming has benefited end-to-end application testing on mobile phones [12]. Robot manipulators have been studied and analyzed for implementing specified control algorithms for efficient usage [13]. Computer vision applications such as image rectification can be tested using programmable robots [14]. Object-tracking-based algorithms that are executed on mobile phones require testing under various relative speeds between the test object and the mobile phone and angular movements. These tests can be executed using a robotic arm [15]. Robots can be used with image processing algorithms for face recognition [16]. Robotic-arm-based 3D reconstruction validation involves verifying the reconstructed 3D objects and executing tests of the algorithmic integrity of the computer vision algorithms that are used for the 3D image reconstruction [17]. The precise movements of a programmable robotic arm can be used to test software features, e.g., in a motion-based image capture system [18]. A 'face plane' from 3D reconstructed face data can be extracted and used for head pose estimation [19].

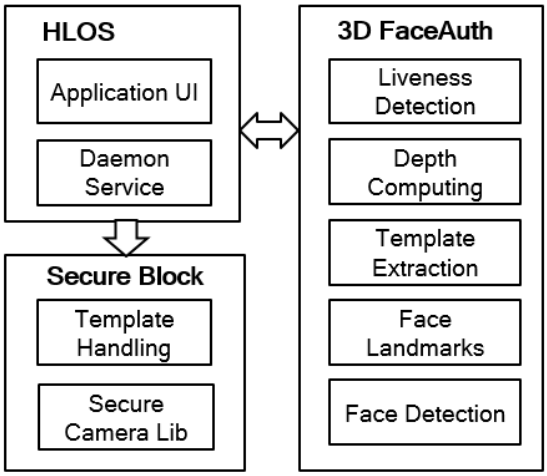


FIGURE 2. 3D face authentication software high-level architecture blocks.

Three-dimensional face authentication algorithms involve face feature extraction and analysis [20]. A face authentication score is generated from each face after comparison with the enrolled faces [21].

For production software systems dealing with a large number of developer code submissions, it is critical to sanitize the quality of software. Thereby a continuous assessment of reliability attained through test execution and analysis [22]. it is necessary to achieve the test coverage levels recommended or mandated by safety standards and industry guidelines [23]. In order to achieve tests in a timely manner software test automation is used to execute test cases in a reliable manner [24]. Testing frameworks can be used to reduce testing efforts by and improve test quality [25]. It is important during the deployment of software test automation any test automation blockers or impediments are monitored and removed in a timely manner to achieve efficient results [26].

Thus, thorough testing is needed with reliable setups for testing the algorithmic integrity of 3D face authentication algorithms.

IV. SOLUTION

A. SOFTWARE OVERVIEW

The 3D face authentication software solution under test consists of three major blocks: The first block is the high-level operation system (HLOS), which contains the 3DFA application user interface. The 3DFA service and daemon are found one layer beneath this block. The 3D face auth-daemon communicates with the 3D face authentication computing block and extracts the face information that is obtained by a secure camera via liveness detection, depth computing, face template extraction, face landmark application and face detection algorithms using a FastRPC bridge. Then, the extracted information is handled by the secure block, which matches the information with the enrolled user face template. Based on the obtained scores and the threshold settings, the software determines whether the authentication is successful. The block diagrams are shown in Fig. 2.

TABLE 1. List of the test cases from the 3DFA basic test requirements.

Item	Condition	Test case
Distance	Distance Indoor light	200 mm
		300 mm
		400 mm
		500 mm
Angle	Pitch	30˚ / -30˚
	Yaw	30˚ / -30˚
	Roll	45˚ / -45˚
Illumination	Indoor light	Lights on
		Without light
	Outdoor light	Daytime front / side /back

B. OBJECTIVE

The main objective of our testing on the 3D face authentication software is to determine the false-rejection rate (FRR) and the latency of the authentication process using only positive face models. False rejection means the end-to-end app cannot authenticate using the enrolled user's face. False-rejection rate is calculated from dividing the number of failed attempts by the numbers of face authentication performed. The 3D face authentication test requirements are selected from three categories: user face unlock distance from the sensors; user face angles with the sensors, which are divided into pitch, yaw and roll angles; and illumination, which includes various indoor and outdoor lighting conditions. All the above conditions have various effects on the software's FRR and latency. The test cases are listed in Table 1.

Since 3D face authentication is a relatively new technology, few application testing methods that are based on this technology have been developed. The traditional method of testing such items is to set the conditions manually and repeat the tests on the software application; however, this method is highly time-consuming and inaccurate. Therefore, our objective is to automate these processes using a robotic arm system to provide more efficient and accurate test results. We can also expand the test cases into additional categories.

C. TEST SETUP

Since our objective is to automate the entire 3D face authentication test, we must find a solution for each testing sequence. The physical motions that are required for end-to-end software testing, such as changing the device distance, angles, and angular motions, can be performed by a robotic arm to simulate a tester's motion in manual testing. We use the Denso robot V87 series, which is a 6-DOF robotic arm that has 6 programmable joints, to more accurately simulate human motions. The robotic arm setup is shown in Fig. 3. Since the 3DFA testing requires a real person's face, the test subject sits in a fixed location next to the robotic arm. The main advantages of using a robotic arm are superior accuracy

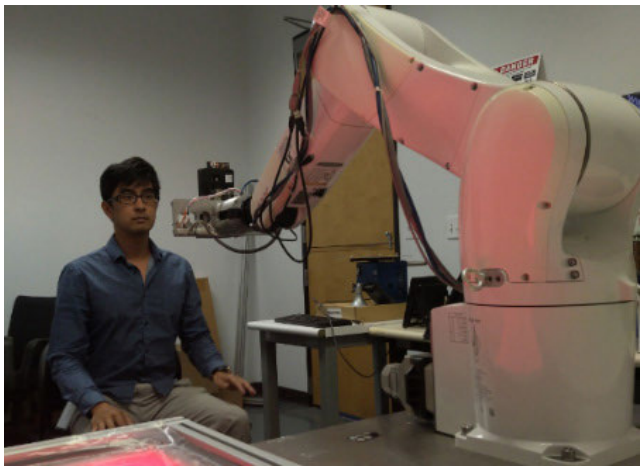


FIGURE 3. Denso robotic arm setup for 3DFA.

and consistency compared with human testing. The robotic arm can be programmed using several compatible software programs and programming languages. We use ORiN2 and WINCAPS3 software to program this robot.

D. GENERAL TEST AUTOMATION WORKFLOW

When executing an automated test case with the robotic arm, a general automation workflow is followed, as illustrated in Fig. 4. First, a test scheduler triggers a 3D face authentication test job that runs a specified use case. The control PC that connects with the robot will pick up the 3D-face-authentication-related job. This test job runs through our main Python automation test script with parameters that are specified by the job's XML. Then, it triggers the VisualBasic executable file with a specified robot program and other test options. The executable file uses ORiN2 as the robot controller middleware to communicate with the controller to run robotic programs that are stored in the controller. The robotic programs directly define the robotic arm's series of motions by turning the 6 joints of the robot to the specified 6-DOF positions. The programming details will be discussed in later sections. When the robotic arm picks up the test device and places it in the initial test position, the main Python script passes a command to the UI Automator, which performs the UI touch events, such as tap, drag and select, name search or UI ID search that a tester would otherwise perform manually. At the same time, the robotic arm moves the test device through various testing positions and motions that simulate a human arm's manual test execution. Upon completion, logs and images are collected from the test device and sent to the control PC. The python script calls the log parser function to extract the result information from the logs. The Python script uses this information to determine the results. Finally, the results are posted to the job link and our database and the job is complete.

E. TEST AUTOMATION FOR VARIOUS AUTHENTICATION DISTANCES

Testing under various distances between the user's face and the camera sensor is the most fundamental 3D face

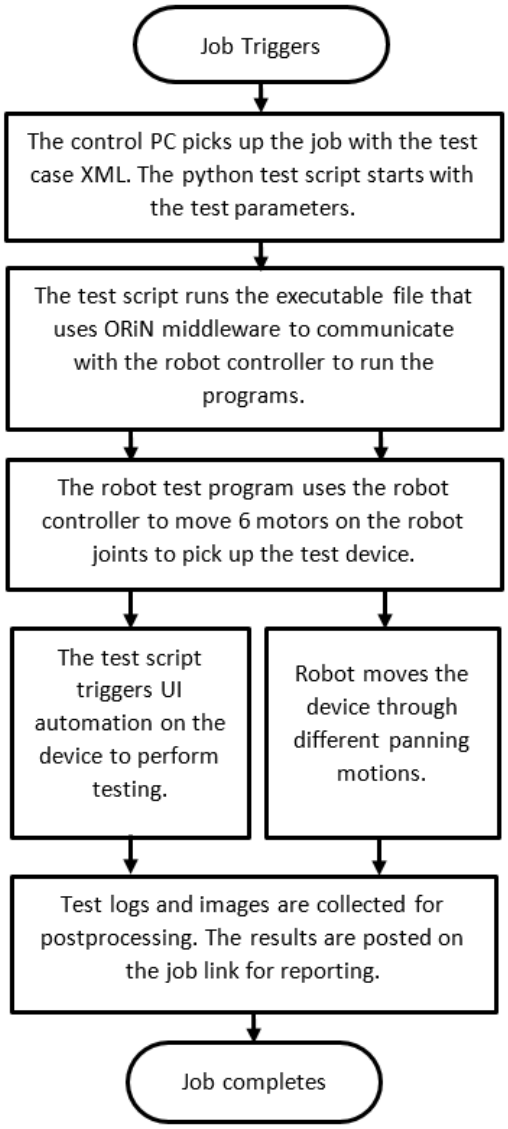


FIGURE 4. 3DFA test automation workflow diagram.

authentication test scenario. Users should be able to successfully authenticate their faces while holding their phones at a comfortable distance. In this case, according to the test requirements, the 3DFA and the camera sensors should support face distances that are between 20 cm and 50 cm. Therefore, we require an automation that verifies 3D face authentication at 20 cm, 30 cm, 40 cm and 50 cm away from the face. First, the test subject must sit at a fixed location in front of the base of the robotic arm. In our case, the test subject sits 100 cm away from the robotic arm and the center of the face is approximately 30 cm above the robot base, as indicated in Fig. 5. Fig. 5 shows a 3D illustration of the robotic arm working environment and provides a reference of the positions in the robotic arm programming.

When programming the robotic arm to hold the device 30 cm away from the face, for example, we must determine the 6-DOF coordinates of the tip of the robotic arm at that position. The 6-DOF coordinate system includes the position

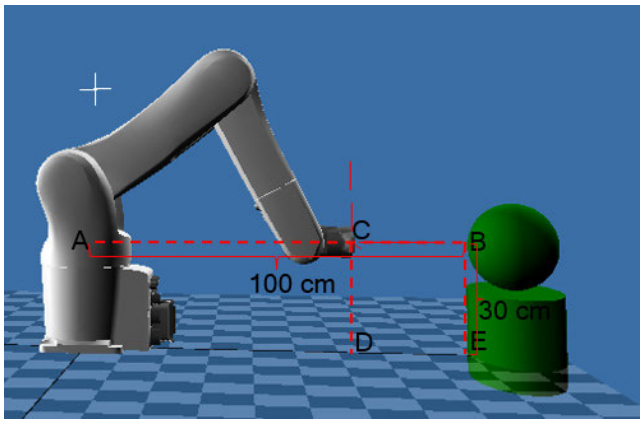


FIGURE 5. Simulation of the robotic arm at 30 cm away from the test subject.

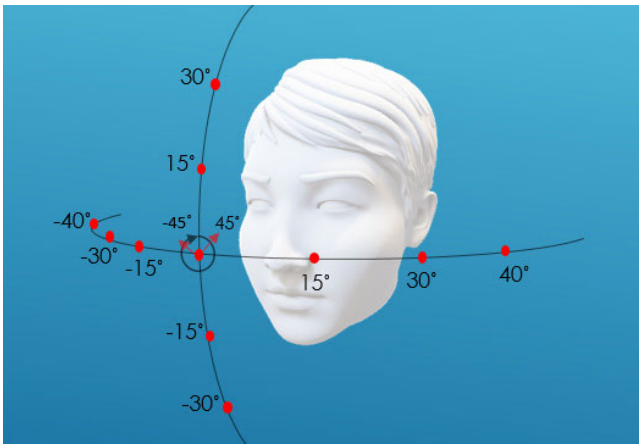


FIGURE 6. Angles and positions for face angular testing.

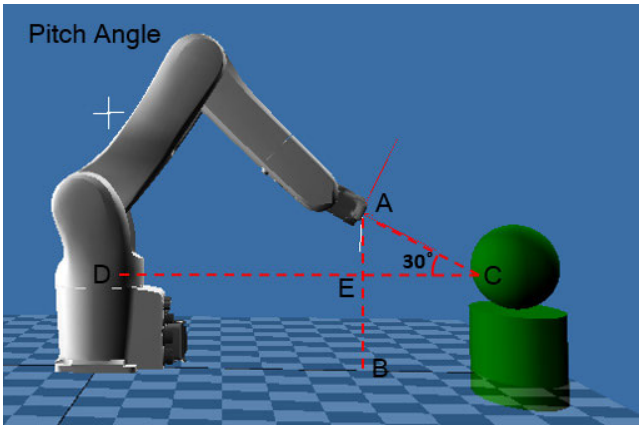


FIGURE 7. 3DFA test automation workflow diagram.

information and the yaw, pitch, and roll rotational information of the tip with respect to the base of the robotic arm in terms of (X, Y, Z) and (RX, RY, RZ) coordinates. First, we must calculate the X-axis coordinate of the tip, which is AC = AB – BC = 100 – 30 = 70 cm. Since the test subject is aligned with the robotic arm's base, the Y-axis coordinate should be 0 cm. The Z-axis coordinate, namely, CD, should be the length of BE = 30 cm. After determining the positional coordinates of the tip of the robotic arm, the next step is to determine the rotational coordinates. Since the tip of the robotic arm directly faces the test subject without yaw, pitch and roll, it is parallel to the XY-plane and rotational coordinates RX and RZ should be 0°. The RY-coordinate should be 90° since the tip is rotated 90° with respect to the Y-axis. Therefore, the 6-DOF position of the robotic arm is determined to be (70, 0, 30, 0°, 90°, and 0°) in the format of (X, Y, Z, RX, RY, and RZ) for a 30 cm distance facing the test subject. Via the same method, we can program the remaining distances for the robotic arm.

F. TEST AUTOMATION FOR ANGULAR POSITIONS SCENARIOS

In 3D face authentication functional and performance testing, testing at various angles of the face is essential. It is critical to determine how the software performs if the face that is used to try to unlock the phone is not directly facing the sensors and where its limits are. Additionally, to calculate the performance metrics, we need to determine how the false-rejection rate and the latency are affected when we push the face angle to its limits. Similar to face recognition testing, we test the 3D face authentication software under various pitch, yaw and roll angles of the test subject's face. According to the software support limits, we test the pitch angles from −30° to 30° with an increment of 15°. The support limits are −30° to 30° for the yaw angle and −45° to 45° for the roll angle. An illustration is presented in Fig. 6. The red dots represent the locations of the testing position device sensors and the corresponding angles are specified.

The robotic arm is highly suitable for this testing scenario. Once the arm has been programmed with the precise location, it is more accurate and consistent than a human tester who is trying to move to these angles. In addition, the robot also saves a substantial amount of time compared to manual testing.

To program the robotic arm to perform yaw, pitch and roll angular movements, we must determine the 6-DOF positions of the robotic arm that will place the device at the 11 points that are marked in Fig. 6. As an example, suppose a pitch angle of 30° is desired. The robotic arm must rise from the position that is illustrated in Figure 6 and form a 30 ° pitch angle with the center of the test subject's face. Since the position of the test subject does not change, the distance, namely, CD, and the height, namely, BE, remain the same (CD = 100 cm and BE = 30 cm). We must calculate the X-axis coordinate, namely, DE, and the Z-axis coordinate, namely, AB, which are illustrated in Figure 7.

Since AB is the sum of AE and BE, where BE is known, and DE is the difference between CD and CE, where CD is known, we must determine the lengths of AE and CE. Points ACE form a right triangle with one angle at 30° and one side, namely, AC, that we want to keep at a distance of 30 cm.

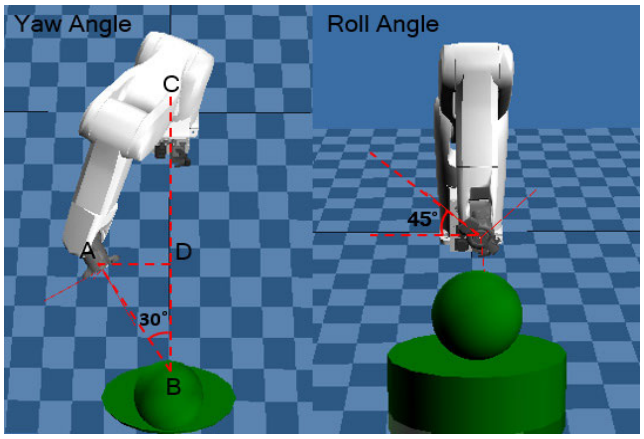


FIGURE 8. Simulation of the robotic arm at a 30° yaw angle and a 45° roll angle.

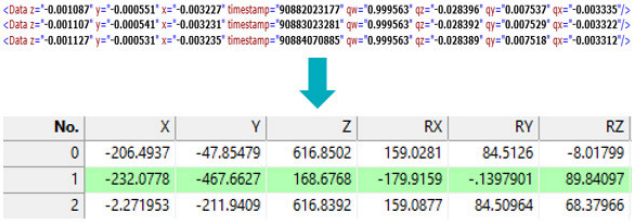
Using trigonometry, the other two sides, AE and CE, can be determined.

$$AE = AC \times \sin 30^\circ = 30 \times 0.5 = 15\ cm$$
$$CE = AC \times \cos 30^\circ = 30 \times 0.866 \approx 26\ cm$$
$$AB = AE + BE = 15 + 30 = 45\ cm$$
$$DE = CD - CE = 100 - 26 = 74\ cm$$

Therefore, the X-axis and Z-axis positional coordinates are determined, and the Y-axis coordinate remains at zero. The next step is to find the rotational coordinates. Since the yaw and roll angles remain at 0° and only the pitch angle changes by 30°, the rotational coordinate for the Y-axis, namely, RY, is reduced by 30° and becomes 90° - 30° = 60°. The robotic arm tip's 6-DOF coordinates for the 30° pitch angle are found to be (74, 0, 45, 0°, 60°, 0°). Then, this robotic arm's 6-DOF coordinates are programmed into the controller and the same procedure can be followed to program the remaining pitch angles of −30°, −15°, and 15°. At each angle, the robotic arm can remain in place temporarily to allow the UI Automator to execute 3D face authentication multiple times before moving on to the next angle.

To realize the various yaw angles, the robotic arm must move the device so that it faces the left and right sides of the subject's face. We maintain the device at the same height as the center of the face and a specified distance from the face. As an example, consider a 30° yaw angle, as shown in the left part of Fig. 8, and the same height (30 cm) and face distance (30 cm) as in the previous example. The Z-axis positional coordinate of the tip of the robotic arm is the height of 30 cm. The X-axis and Y-axis coordinates must be calculated. As shown in the diagram, point ABD forms a right triangle. The length of CD is the X-axis coordinate and AD is the Y-axis coordinate.

$$AD = AB \times \sin 30^\circ = 30 \times 0.5 = 15\ cm$$
$$BD = AB \times \cos 30^\circ = 30 \times 0.866 \approx 26\ cm$$
$$CD = BC - BD = 100 - 26 = 74\ cm$$



No.	X	Y	Z	RX	RY	RZ
0	-206.4937	-47.85479	616.8502	159.0281	84.5126	-8.01799
1	-232.0778	-467.6627	168.6768	-179.9159	-.1397901	89.84097
2	-2.271953	-211.9409	616.8392	159.0877	84.50964	68.37966

FIGURE 9. 6-DOF coordinates mapping for simulating real-life device motions.

For the rotational coordinates, since the pitch angle is 0°, the coordinate RY remains 90°. The yaw angle is set to 30°; thus, RX should also be 30°. RZ remains at 0°. Therefore, the robotic arm tip's 6-DOF coordinates should be (74, 15, 30, 30°, 90°, and 0°) for a 30° yaw angle with the face of the test subject. The same calculation method is applied for the remaining yaw angles.

For programming the roll angles, the device is positioned facing the test subject's face. Then, the device rotates clockwise or counterclockwise to the specified roll angles. This rotation can be easily realized with the robotic arm starting at the standard position facing the test subject, as explained previously. Then, the tip joint – joint 6 – of the robotic arm is rotated to realize the desired roll angle. In this case, it is a 45° roll angle, as shown in the right part of Fig. 8.

Therefore, the robotic arm can be used to perform all the above face angle requirement tests. Once the positions have been programmed and measured, the angular 3D face authentication tests can be executed and repeated with much higher efficiency and accuracy compared to manual testing.

G. TEST AUTOMATION FOR DEVICE MOTION SCENARIOS

In real-life scenarios, the most common use case of 3D face authentication is the face unlock feature for smartphones. Users may not always unlock their phones in a stationary manner. Users may try to use their faces to unlock their phones when pulling them out of pockets or looking at the phones while they are raised to face level. By this time, the phones should already be unlocked. A user may be walking or running while attempting to use the software or may be sitting next to a table and pick up the phone from the left- or right-hand side, such that the user's arms move the phone to his or her face to unlock it. In all these scenarios, face authentication is performed while the device is in motion. We need to closely simulate these 6-Dof motions using the robotic arm. First, the 6-Dof coordinates of the device in motion need to be captured while the tester manually performing these face authentication motions. This positional information can be collected using the device's gyroscope and accelerometer. The coordinates are usually saved in quaternion format (z, y, x, qw, qz, qy, qx) based on time stamps. Then, we need to sample these coordinates and convert them into robotic arm system's 6-Dof coordinates (X, Y, Z, RX, RY, RZ) as demonstrated in figure 9. A proper conversion requires the alignment of the 3-dimenstional coordinates of device and

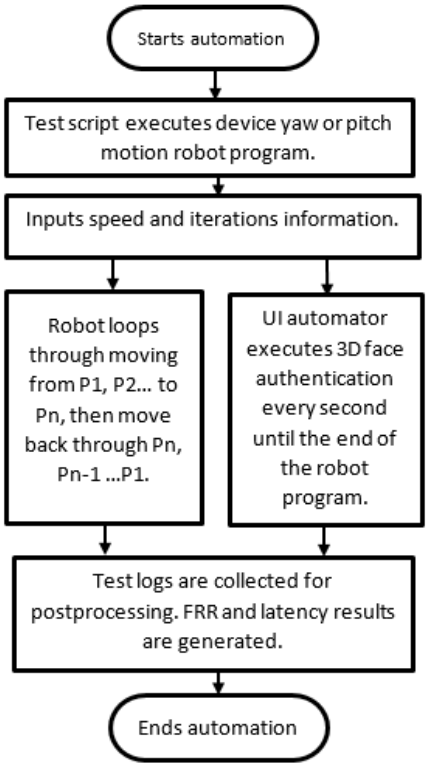


FIGURE 10. Workflow of the device motion test automation.

the base of the robotic arm, as well as the rotational angle system calculation. For example, the angular coordinates conversion can be performed using Matlab software function: $[r_1, r_2, r_3] = quat2angle(q)$, where $q = (qw, qz, qy, qx)$.

Finally, after obtained the necessary robot system coordinates, we can program the robotic arm to move through these points to simulate device motions like device jitter while user is running and pulling the device out of pocket.

When testing 3D face authentication in motion, the motion speed is a factor that will affect the success rate and the authentication latency. Therefore, the test can be performed under various motion speeds to study the effects. Since the robotic arm's speed can be set by its controller, we can program the robotic arm to run the motions with different speeds. The 3D face authentication tests are performed in parallel while the robotic arm performs the device motion, as shown in the workflow diagram in Fig. 10.

After the robotic program has finished all the iterations, the 3D face authentication execution stops on the test device. The test logs and data are collected for postprocessing to calculate the false-rejection rate from the authentication during the device motion; the face recognition scores, 3D liveness scores and latency results are also captured for comparative analysis.

H. TESTING ON VARIOUS FACIAL APPEARANCES

In our 3D face authentication test plan, we covered many other real-life scenarios, such as the various facial appearances of the enrolled users. We seek to ensure that users can still unlock their device using their faces if they are wearing different accessories or have slightly different appearances each day. The first column of Table 2 lists common variations in a user's appearance.

TABLE 2. Test scenario table for various appearances, expressions and lighting conditions.

Appearance	Expression	Lighting Condition
Mustache/Beard	Smiling	Indoor normal
Makeup	Laughing	Indoor low light
Hairstyle	Surprise	Indoor bright light
Hat/Cap	Sad	Indoor colored light
Helmet	Angry	Outdoor night light
Glasses	Sleepy	Outdoor bright light

Test subjects enroll themselves before applying the facial changes and the accessories. When conducting the 3D face authentication tests, the subjects can wear hats, caps, various types of helmets, thick-frame reading glasses and sunglasses to try to alter their appearance. These appearance changes are easy to implement and return valuable test results. These changes can also be used in face model training.

In addition, the enrolled users are asked to vary their facial expressions during the 3D face authentication. Expressions such as smile, laughter, surprise, sadness, anger and sleepiness are listed in the second column of Table 2. The difference between smile and laugh is laugh shows teeth with mouth open while smile doesn't. Surprise expression will have the user with eyes and mouth wide open, while sleepy expression requires the user open the eyes halfway. These expressions cause the user's appearance to differ from the enrolled appearance. The differences in the expressions challenge the facial landmarks and face recognition algorithm's ability to identify the key features of the user. If the face recognition scores drop below the threshold score, it will not authenticate the user correctly and therefore reduces authentication accuracy. Our test automation for this use case only requires the user to sit in front of the robotic arm. Our test script includes a voice guidance feature that tells the user which expressions to make during the test, and an image chart of each expression that the user can follow.

Finally, various lighting conditions are also key scenarios for testing 3D face authentication and have always been part of the test requirements. As listed in the third column of Table 2, we test the common lighting conditions that users may encounter during 3D face authentication. These include normal light, low light, bright light, colored light and no light in an indoor scenario and bright light, side light, low light and night light in an outdoor lighting scenario. We simulate the indoor lighting conditions by installing multiple controllable light sources in the robotic arm lab. By programming the light sources to turn on/off and changing the brightness, we can control the light intensity in the testing environment, as shown in Fig. 11.

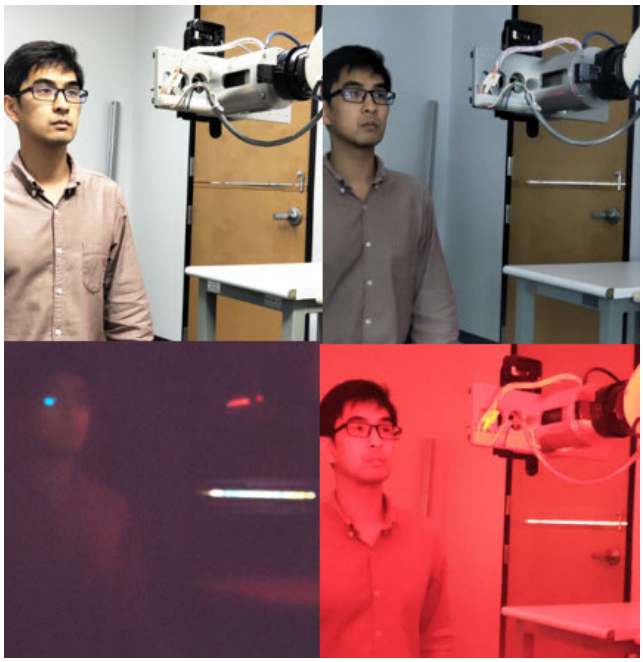


FIGURE 11. Indoor test environment with bright light, low light, no light and colored light conditions.

Even though we have full control of the indoor test lighting conditions, we are unable to simulate the outdoor conditions in the lab. The tests are executed manually outside to obtain the most realistic results.

FIGURE 12. a. Snapshots from yaw angle position tests. b. Snapshots from pitch angle position tests. c. Snapshots from roll angle position tests.

V. RESULTS

A. RESULTS FROM THE ANGULAR POSITION SCENARIO TESTS

We initiated the angular 3D face authentication test by recruiting 10 people as test subjects, and we ran the entire angular test sequence 10 times for each person. Thus, a total of 100 results were collected for each angular position. Fig. 12a shows sample snapshots from the frontal position and the 15°, 30° and 40° yaw angle positions. Fig. 12b shows sample snapshots from the ±15° and ±30° pitch angle positions. In addition, Fig. 12c shows the sample snapshots from the 15°, 30° and ±45° roll angle positions.

In this angular 3D face authentication test, we measured face angular position impacts on the authentication latency, the accuracy (false-rejection rate), as well as 2D face recognition scores and the 3D liveness scores by leaving all other conditions that same and only changing the face angles. The face recognition score measures how close the face features extracted from the frames compare to the enrolled face templates. The 2D face recognition score ranges from 0 to 1.0, and the threshold for accepting a face is 0.7 in the application under test. The 3D liveness score measures how well the mesh represent a 3D face. The range is from 0 to 1.0 and threshold for accepting as real face is 0.8. Finally, all the test results were averaged and listed in Table 3.

According to this table, at the frontal position, the average latency was 200 ms; we used this as the base latency. The false-rejection rate was 0%; hence, all the authentication attempts succeeded at the frontal position. As the pitch angle increased, the latency increased by 24.5% at 15° pitch up and down and by 77.5% at 30° pitch angles. Additionally, at 30° pitch angles, the 1% FRR demonstrated that the 3D face authentication might start failing beyond ±30° pitch angles. However, the results also demonstrated that these pitch angles were not yet at the limit of what this 3D face authentication software could support. At yaw angles of ±15°, the average latency increased to 22% similar to the pitch angles. The FRR was already at 2%. At ±30° yaw angles, the latency increased to 117.5%, which was significantly higher than the increase rate of the pitch angles. The FRR became 24%; hence, at ±30° yaw angles, we could no longer consistently authenticate faces. In addition, at ±40° yaw angles, we were obtaining a 247% increase in latency and the software was having difficulty authenticating the faces since almost half of the faces were hidden. The FRR was 72%; hence, the 3D face authentication had reached the support limit and timed out and most of the time, it was unable to recognize the faces at ±40° yaw angles. As the roll angles increased, the latency increased from 15.5% at ±15° to 86% at ±30° angles. At ±45° roll angles, the latency increased 250%, similar to the yaw angles at ±40°. The FRR increased by 2% for every 15° increase in the angle. In addition, the 2D face recognition scores follow the same trend as the FRR. When the average face recognition score fell below 0.7, the false

TABLE 3. Latency and accuracy results under various face angles.

Latency and Accuracy Results for Various Face Angles			
Face Angle	**Frontal**	**Pitch ± 15°**	**Pitch ± 30°**
Latency	200 ms	249 ms	355 ms
FRR	0%	0%	1%
Face Recognition Score	0.91	0.85	0.78
3D Liveness Score	0.99	0.95	0.93
Face Angle	**Yaw ± 15°**	**Yaw ± 30°**	**Yaw ± 40°**
Latency	244 ms	435 ms	694 ms
FRR	2%	24%	72%
Face Recognition Score	0.75	0.69	0.65
3D Liveness Score	0.95	0.92	0.91
Face Angle	**Roll ± 15°**	**Roll ± 30°**	**Roll ± 45°**
Latency	231 ms	372 ms	700 ms
FRR	0%	2%	4%
Face Recognition Score	0.85	0.77	0.72
3D Liveness Score	0.96	0.94	0.93

rejection rate increased significantly. Finally, the 3D liveness scores stayed above 0.9 since we used real user's face in this experiment. The test result data were plotted in the bar graphs in Fig. 13a, Fig. 13b and Fig. 13c.

According to the bar graph, the 3D face authentication latency grew exponentially as the face angles increased. Additionally, the authentication accuracy was most sensitive to changes in the face yaw angles; changes in the pitch and roll angles had smaller effects on the accuracy in the lower angle range.

B. RESULTS FROM DEVICE MOTION TESTS

In the device motion automated testing experiment, we designed the robotic arm to perform device authentication under running motion simulation with low, medium and high device motion speeds. For each speed level, the automation executed 3D face authentication 100 times. This test was repeated with a 3rd-party face unlock solution on another device to study and compare the effects of device motion on the face authentication latency and accuracy. The results were collected and averaged from both solutions after the automated tests were completed. The latency, accuracy, 2D face recognition score and 3D liveness score results are listed in Table 4.

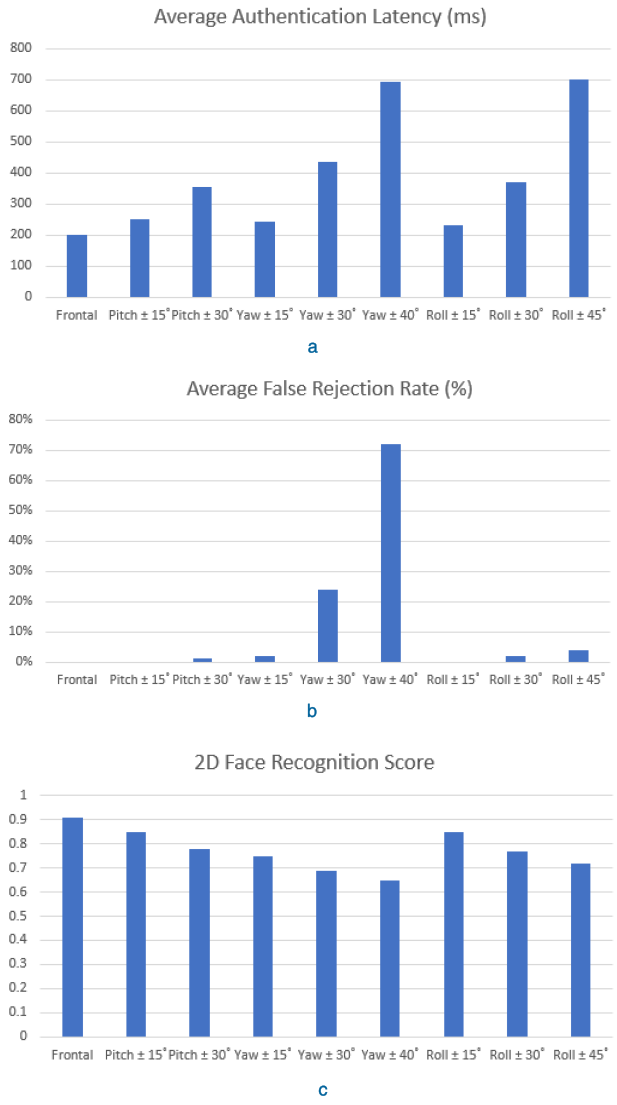


FIGURE 13. a. Bar graph of the average latencies at nine face angles. b. Bar graph of the false-rejection rates at nine face angles. c. Bar graph of the face recognition scores at nine face angles.

According to this result table, the face authentication latency began to increase as the motion speed increased. Comparing the 3DFA application and the 3rd-party application latency results, the 3DFA application had lower latency than the 3rd-party application on all the motion speeds in the table. This result could be due to the 3DFA application used PMD 3D sensors, which captured clearer depth images than the 3rd-party application's sensors. In addition, the combination of our optimized 3D face authentication software algorithm and the accelerated-hardware solutions resulted in an overall face authentication time that was 23.9% faster under this device's motion test scenario. A comparison bar graph is shown in Fig. 14 to facilitate visualization of the data.

As shown in the bar graph, at lower speed, the latency increased more slowly, and at higher speed, the latency

TABLE 4. Device motion experiment on 3D face authentication applications test result comparison table.

Device Motion Latency and Accuracy Comparison Results			
Device Motion Speed	**Low**	**Medium**	**High**
3DFA App Latency	233 ms	343 ms	515 ms
3rd-party App Latency	362 ms	435 ms	636 ms
3DFA App FRR	0%	1%	2%
3rd-party App FRR	1%	3%	5%
3DFA App Face Recognition Score	0.85	0.83	0.80
3rd-party App Face Recognition Score	0.80	0.77	0.73
3DFA App 3D Liveness Score	0.92	0.90	0.89
3rd-party App 3D Liveness Score	0.91	0.88	0.86

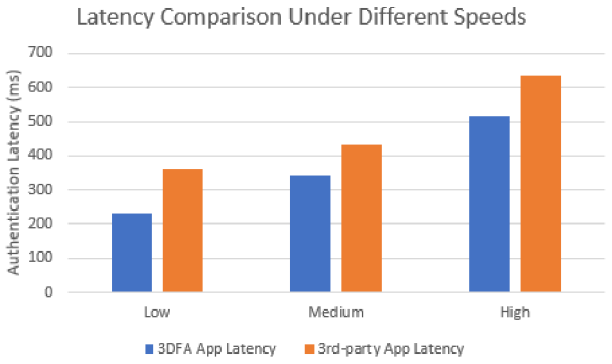


FIGURE 14. Bar graph for the yaw and pitch device motion scenario latency comparison.

increased more rapidly. This result could be caused by the blurriness in the face image that was captured for face authentication, which was induced by the device motion. At this latency growth rate, we anticipated that at high enough device motion speeds, there would be a large performance impact on the latency, which could cause the authentication process to time out and eventually fail. However, such high-frequency device motion scenarios rarely occur in daily usage.

From the face authentication accuracy performance perspective, the FRR increased slightly as the device motion speed increased. The 3rd-party application also had slightly higher FRR than the 3DFA application. Therefore, the 3rd-party application had higher chance to fail the face authentication while the device was in motion. On the other hand, the 2D face recognition scores and 3D liveness scores were not affected much by the device motion. Since these score metrics were only depend on the user's face, and the user's face had little variation during device's motion, if the frames captured had the complete user's face inside, the scores should remain unaffected.

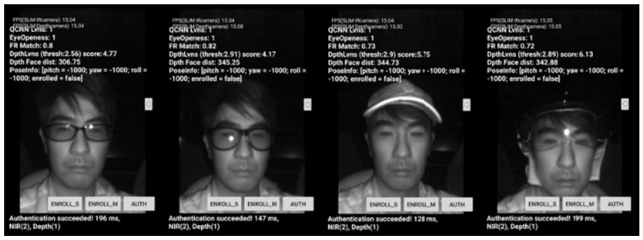


FIGURE 15. Sample snapshots with the test subject wearing reading glasses, sunglasses, a cap and a helmet.

C. RESULTS OF THE FACIAL-APPEARANCE SCENARIO TESTS

During the facial-appearance automated testing, three main tests were conducted, in which the following characteristics were varied: accessories, expressions and lighting conditions. These 3D face authentication tests required a real person to be the test subject. We again invited 10 people to be our test subjects.

In the first test, all the test subjects were enrolled in the 3D face authentication software. Then, they put on accessories that we prepared as part of the setup to change their appearances, including fake mustaches, wigs, caps, hats, helmets, reading glasses and sunglasses. The robotic arm placed the test device in the standard frontal position while the automation authenticated their faces. Sample snapshots from the 3D face authentication testing with accessories are shown in Fig. 15.

The 3D face authentication software was able to correctly authenticate the test subjects when they were wearing mustaches, wigs, caps and hats. These accessories did not change the appearance of the key features of the faces. The software also performed well on the test subjects when they were wearing glasses since the face model that the software used was trained to recognize glasses. Additionally, the near-infrared sensor was able to penetrate most lenses of the sunglasses, as shown in the second image in Fig. 16, in which the areas of the face are clearly visible.

The second test required the test subjects to make various expressions to change their appearance. The robotic arm placed the device in the frontal position and the automated test system used voice guidance and expression chart to tell the test subjects what expressions to make. Fig. 16 shows sample images from the 3D face authentication test application with various facial expressions.

Since the test subjects were in the frontal position and only changed their expressions, we expected them to pass the depth and liveness tests. The main variation would be in the face recognition scores because extreme expressions might make the faces look substantially different from the enrolled faces. This test was intended to challenge the face recognition algorithm, which is part of the 3D face authentication process. We parsed the logs and collected the face recognition scores from each run; the average scores are listed in Table 5.

The face recognition score quantified the similarity between the input face profile and the enrolled face profile.

FIGURE 16. Sample result images with six facial expressions: laugh, smile, surprise, sad, angry and sleepy.

TABLE 5. Average face recognition scores for six expressions.

Face Recognition Scores and Latency for Facial Expressions						
Expressions	smile	laugh	surprise	sad	sad	sleepy
Latency	223 ms	243 ms	247 ms	227 ms	262 ms	224 ms
FR Scores	0.83	0.77	0.75	0.83	0.72	0.84

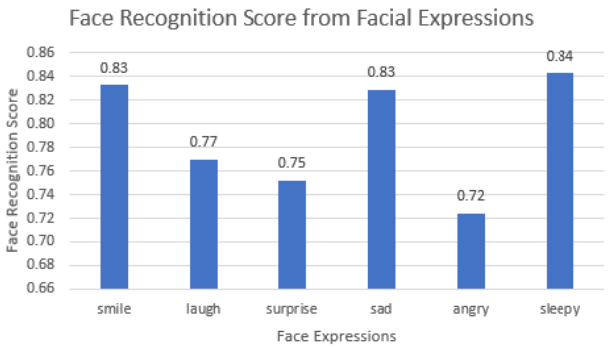


FIGURE 17. Bar graph of the face recognition scores for six facial expressions.

The face recognition score threshold was set to 0.7; faces whose scores exceeded this threshold passed the test. Since all the scores for faces with the various facial expressions exceeded 0.7, they all passed the face recognition check process. However, according to the bar graph in Fig. 17, the scores varied among the expressions.

The smile, sad and sleepy expressions corresponded to face recognition scores of approximately 0.83. Hence, when making these expressions, the facial features still matched the neutral enrolled expression well, but with lower eye openness scores. The laugh, surprise and angry expressions corresponded to scores that were between 0.72 and 0.77, which were noticeably lower because these expressions are more extreme and the mouth was open in these cases. The facial features deviated further from the enrolled expression. The results of this test demonstrated that common expressions were able to pass the face recognition test and the overall 3D face authentication test. Certain extreme expressions might take more frames to find a face match resulted in slightly higher face authentication latency. According to table 5, most expressions had averaged latency within 250 ms which were not far from the average latency of 200 ms without expressions. Therefore, this 3D face authentication application is suitable for use on faces with expressions.

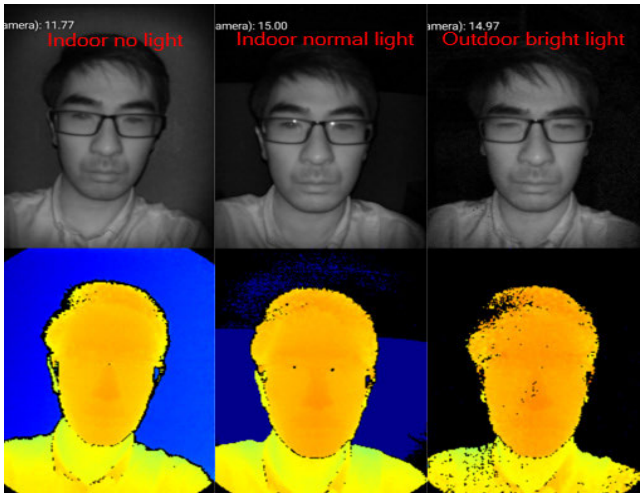


FIGURE 18. Face images that were obtained by the camera sensors under various lighting conditions.

The third test that we conducted was 3D face authentication under various lighting conditions. Our robotic lab used a multiple-dimmable-light setup that simulated various indoor lighting conditions. Since the 3D face authentication software used a near-infrared sensor, the test subjects' faces were clearly visible in the NIR and depth maps, even in a dark room, as shown in Fig. 18. The images in the middle were obtained by the NIR and depth sensors under normal indoor light conditions and are shown for comparison. The images on the right were captured outdoors under direct sunlight. We still obtained clear NIR images under this bright-light condition. In the depth images, there was slight interference from the sunlight in collecting the emitted LED lights that were reflected by the test subject. These illumination tests depended heavily on the camera sensors that were used on the test device for obtaining good images under the extreme lighting conditions. Overall, the 3D face authentication tests were passed in all the indoor and outdoor lighting conditions.

D. COMPARISON WITH OTHER TEST METHODS

Since the advance in technology, 3D face authentication and face unlock become a secure biometric identification option. Many researches have been done on the 3D face

authentication software algorithms and the validation of algorithms on 3D datasets. However, there are very little research on testing the end-to-end 3D face authentication applications in the field. Most of the application level testing are still conducted manually. Survey based 3D face authentication application data collection and performance evaluation are most common. Volunteers are told to hold the devices with applications at different angles and posts. Testers have to manually change lighting conditions or move to outdoor scenes. These manual testing and data collection method are error-prone and time consuming depend on how familiar the users are with operating the application. Our purposed automated testing method takes complete control of the test device and the 3D face authentication application. The users only need minimal involvement. And the automation provides exceptional motion accuracy and efficiency. The entire set of performance evaluation tests using the automation system takes less than half the time conducted in a manual survey.

Some other test methods involved unit testing where canned 3D face data is used to bypass camera sensors for face authentication. This test method can verify the software algorithm performance without user involvement at testing stage. However, the camera sensor calibration used for user data collection is very sensitive to the trained model file and the test app used for unit testing. It is unlikely that user data from an online 3D face dataset will work for the enrollment and authentication of a unit test app. On the other hand, our end-to-end test automation system will ease the users face data collection process under different test scenarios. The user data can be collected efficiently with high precision and consistency for both end-to-end application testing and offline unit testing.

VI. CONCLUSION

The development of the 3D face authentication test automation has helped us improve the testing efficiency. We considered various test scenarios and collected test results from facial-appearance scenario tests, device motion tests, and angular position scenario tests on a 3DFA end-to-end application. We also used this automated test system to compare the hardware-accelerated solution of 3DFA with 3rd-party solutions and observed a 23.9% difference in the latency for face authentication under user running motion test scenarios.

The test automation is proven to be more reliable and efficient than manual face authentication testing. This test system can be easily used to benchmark various performance attributes across multiple 3D face authentication software application products. It helps to study the strengths and weaknesses of each product under different performance impacting scenarios. We are continue working on developing new test automations with robotic arm testing 3D face authentication applications as they evolve with software optimization, enhanced camera sensors and new hardware designs. New test gates and metrics can be added for more comprehensive 3D face authentication end-to-end application testing.

REFERENCES

[1] S. Omidshafiei, A.-A. Agha-Mohammadi, Y. Fan Chen, N. K. Ure, S.-Y. Liu, B. T. Lopez, R. Surati, J. P. How, and J. Vian, "Measurable augmented reality for prototyping cyberphysical systems: A robotics platform to aid the hardware prototyping and performance testing of algorithms," *IEEE Control Syst. Mag.*, vol. 36, no. 6, pp. 65–87, Dec. 2016.

[2] J. P. How, B. Behihke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Syst. Mag.*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[3] Y. Li, S. Li, Q. Song, H. Liu, and Q.-H. M. Meng, "Fast and robust data association using posterior based approximate joint compatibility test," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 331–339, Feb. 2014.

[4] R. C. Luo and C.-C. Chang, "Multisensor fusion and integration: A review on approaches and its applications in mechatronics," *IEEE Trans Ind. Informat.*, vol. 8, no. 1, pp. 49–60, Feb. 2012.

[5] S. Ergur and M. Ozkan, "Trajectory planning of industrial robots for 3-D visualization a ROS-based simulation framework," in *Proc. IEEE Int. Symp. Robot. Manuf. Automat. (ROMA)*, Kuala Lumpur, Malaysia, Dec. 2014, pp. 15–16.

[6] R. B. Rusu and S. Cousins, "3D is here: Point cloud library (PCL)," in *Proc. IEEE Int. Conf. Robot. Autom.*, Shanghai, China, May 2011, pp. 1–4.

[7] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3D registration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Kobe, Japan, May 2009, pp. 3212–3217.

[8] R. B. Rusu, N. Blodow, Z. C. Marton, and M. Beetz, "Aligning point cloud views using persistent feature histograms," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2008, pp. 3384–3391.

[9] R. Awad, G. Heppner, A. Roennau, and M. Bordignon, "ROS engineering workbench based on semantically enriched app models for improved reusability," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Berlin, Germany, Sep. 2016, pp. 1–9.

[10] Y. Hua, S. Zander, M. Bordignon, and B. Hein, "From AutomationML to ros—A model-driven approach for software engineering of industrial robotics using ontological reasoning," in *Proc. IEEE 21st Conf. Emerg. Technol. Factory Automat. (ETFA)*, Sep. 2016, pp. 1–8.

[11] D. Banerjee and K. Yu, "Robotic arm-based face recognition software test automation," *IEEE Access*, vol. 6, pp. 37858–37868, 2018.

[12] D. Banerjee, K. Yu, and G. Aggarwal, "Hand jitter reduction algorithm software test automation using robotic arm," *IEEE Access*, vol. 6, pp. 23582–23590, 2018.

[13] Y. Wang, L. Gu, Y. Xu, and X. Cao, "Practical tracking control of robot manipulators with continuous fractional-order nonsingular terminal sliding mode," *IEEE Trans. Ind. Electron.*, vol. 63, no. 10, pp. 6194–6204, Oct. 2016.

[14] D. Banerjee, K. Yu, and G. Aggarwal, "Image rectification software test automation using a robotic ARM," *IEEE Access*, vol. 6, pp. 34075–34085, 2018.

[15] D. Banerjee, K. Yu, and G. Aggarwal, "Object tracking test automation using a robotic arm," *IEEE Access*, vol. 6, pp. 56378–56394, 2018, doi: 10.1109/ACCESS.2018.2873284.

[16] R. Bormann, T. Zwolfer, J. Fischer, J. Hampp, and M. Hagele, "Person recognition for service robotics applications," in *Proc. 13th IEEE-RAS Int. Conf. Humanoid Robots (Humanoids)*, Atlanta, GA, USA, Oct. 2013, pp. 15–17.

[17] D. Banerjee, K. Yu, and G. Aggarwal, "Robotic arm based 3D reconstruction test automation," *IEEE Access*, vol. 6, pp. 7206–7213, 2018.

[18] D. Banerjee and K. Yu, "Integrated test automation for evaluating a motion-based image capture system using a robotic arm," *IEEE Access*, vol. 7, pp. 1888–1896, 2019, doi: 10.1109/ACCESS.2018.2886272.

[19] S. Gurbuz, E. Oztop, and N. Inoue, "Model free head pose estimation using stereovision," *Pattern Recognit.*, vol. 45, no. 1, pp. 33–42, 2012.

[20] N. Uchida, T. Shibahara, T. Aoki, H. Nakajima, and K. Kobayashi, "3D face recognition using passive stereo vision," in *Proc. IEEE Int. Conf. Image Process.*, Genova, Italy, Nov. 2005, p. 950.

[21] C. C. Queirolo, L. Silva, O. R. P. Bellon, and M. P. Segundo, "3D face recognition using simulated annealing and the surface interpenetration measure," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 2, pp. 206–219, Feb. 2010.

[22] D. Cotroneo, R. Pietrantuono, and S. Russo, "RELAI testing: A technique to assess and improve software reliability," *IEEE Trans. Softw. Eng.*, vol. 42, no. 5, pp. 452–475, May 2016.

[23] R. Baker and I. Habli, "An empirical evaluation of mutation testing for improving the test quality of safety-critical software," *IEEE Trans. Softw. Eng.*, vol. 39, no. 6, pp. 787–805, Sep. 2012.

[24] P. E. Strandberg, E. P. Enoiu, W. Afzal, D. Sundmark, and R. Feldt, "Information flow in software testing—An interview study with embedded software engineering practitioners," *IEEE Access*, vol. 7, pp. 46434–46453, 2019.
[25] M. Becker, D. Baldin, C. Kuznik, M. M. Joy, T. Xie, and W. Mueller, "XEMU: An efficient QEMU based binary mutation testing framework for embedded software," in *Proc. 10th ACM Int. Conf. Embedded Softw. (EMSOFT)*, 2012, pp. 33–42.
[26] K. Wiklund, S. Eldh, D. Sundmark, and K. Lundqvist, "Impediments for software test automation: A systematic literature review," *Softw. Test., Verification Rel.*, vol. 27, no. 8, p. e1639, Sep. 2017.

DEBDEEP BANERJEE (Member, IEEE) received the M.S. degree in electrical engineering from the Illinois Institute of Technology.

He has more than 11 years of industry experience in the field of software/systems engineering. He is the software/systems development engineer test lead for the computer vision project and is responsible for test automation design, planning, development, deployment, code reviews, and project management. He works closely with the software/system teams. He also works in the graphics software test development for machine learning and Vulkan projects. He has been working with the software test automation team since the inception of the computer vision project at Qualcomm and is involved in managing and developing the robotic arm software that is used in the Computer Vision Laboratory. He is currently a Senior Staff Engineer and an Engineering Manager with Qualcomm Technologies, Inc., USA. He is also the author of seven published IEEE engineering articles.

KEVIN YU was born in Shenyang, Liaoning, China, in 1987. He received the B.S. degree in electrical engineering from the University of California, San Diego, in 2013.

He is currently a Senior Test Engineer with Qualcomm Technologies, Inc., USA. He has contributed to test automation validation for continuous integration and regression tests for computer vision algorithms. He has worked on robotic arm test automations for computer vision software testing. He is also the author of seven published IEEE engineering articles.