1 BUG Classification

Based on the different bugs we observed in our manual tests, we identified six bugs based on floating windows.

- Penetration Phenomenon
- Camera Call Conflicts
- Recording Resource Conflicts
- Video Playback Conflict
- App Black Screen Exception when Recording Screen
- Keyboard Problems

These exceptions not only affect the experience of cell phone users under HongMeng system, but also bring great security risks. In the following, we will define, analyze and demonstrate the six kinds of exceptions respectively, and list the APPs with such exceptions in manual testing.

1.1 Penetration Phenomenon

Penetration is a phenomenon in which an app interface is opened in FF(Free-Form) or PIP(Picture-In-Picture) mode as A, and an app interface is opened in the main interface as B. Successive clicks on a component of A will not only cause the component itself to respond, but also cause the component to respond to a component of B in the corresponding location on the screen. This anomaly behaves as if the user has clicked through the floating window at the top of the screen (interface A) and clicked on other components below (interface B) and responded to them, so it is figuratively called the "penetration phenomenon".

In order to test the penetration phenomenon, we wrote a test program for video playback a and added a function to enter PIP mode, adding a penetration test button in PIP mode and displaying the test count by clicking the test button continuously. A test program b was written and a large enough button was added to implement the page jumping function to ensure that it could be clicked and jumped to a new page when the top floating window was penetrated after the main interface was opened. In the example, we use program a and program b to test as many times as possible, and only once we found that the penetration phenomenon, i.e., when the test button in a is clicked, the lower layer of b will jump to the page, which shows that in the multi-view state, the penetration phenomenon is rare but does exist.

1.2 Camera Call Conflicts

A camera call conflict is an abnormal situation where an app is opened in FF(Free-Form) mode or PIP (Picture-In-Picture) mode as A, while an app is opened in the main interface as B. When A and B need to use the camera resources at the same time, the app that gets the focus is unable to get the camera resources (in time).

To facilitate the description, we define several concepts that appear in this section.

1. gain focus: an app is in the main interface or when the user actively clicks on the app, we say the app gains focus.
2. gain camera resources: when multiple applications need to use the camera at the same

time, due to resource constraints, only one application's camera can be turned on and used normally, the other applications camera screen is in a black screen or stagnant, we say this application gains camera resources, the other applications lose camera resources.
3. Camera options: In conference applications, users can control the camera options to achieve the purpose of turning on and off the camera.

The camera call conflict issues are classified into 4 subcategories based on the observed cases.

- Unable to get camera resources after PIP mode focus switch
- Unable to get camera resources after FF mode focus switch
- Delayed access to camera resources after focus switch
- Camera option automatically turned off

1.2.1 Camera resources not available after PIP mode focus switch

Open an APP as A in FF(Free-Form) mode or PIP (Picture-In-Picture) mode, and open an APP as B in the main interface, A and B need to use camera resources at the same time and B gets the focus, then switch the relationship between A and B, that is, make A open in the main interface and B open in PIP mode, then A gets the focus, the expected normal situation is --A gets the camera resource, but in practice an exception occurs --A cannot get the camera resource.

In the example demonstration, we use QQ and Feishu, two common videos calling and conferencing APPs, for illustration and demonstration. Firstly, open video call on QQ and make it enter PIP mode, QQ get focus, verify QQ get camera resources, no exception; then open Feishu in the main interface for video conference, at this time Feishu get focus, verify Feishu get camera resources, QQ lose camera resources, no exception; finally, let Feishu enter PIP mode, click QQ small window to make it resume the main interface, at this time QQ Finally, let Feishu enter PIP mode, click QQ small window to restore the main interface, then QQ gets the focus, verify that QQ does not get the camera resources, the camera resources are still held by Feishu, an exception occurs, that is, after the focus switch in PIP mode but can not get the corresponding camera resources.

1.2.2 Can't get camera resources after FF mode focus switch

Similar to subclass 1, after swapping the relationship between A and B, A opens in the main interface, B can choose to open in PIP mode or just close B. At this time A gets the focus, the expected normal situation is - A gets the camera resource, but in practice an exception happens - -A cannot get the camera resource.

In the example demo, we use QQ and Fishu, two common video calling and conferencing APPs, for illustration and demonstration. First, open video conference on Fishu, Fishu get focus, verify Fishu get camera resources, no exception; then open QQ in FF mode and make video call, at this time QQ get focus, verify QQ get camera resources, Fishu lose camera resources, no exception; finally, close QQ and click Fishu meeting in the home screen state, at this time Fishu get focus, verify find Fishu does not Get the camera resources, an exception occurs, that is, after switching the focus in FF mode but can not get the corresponding camera resources.

1.2.3 Delayed acquisition of camera resources after focus switch

Also similar to subclass 1, we expect the normal case to be - A gets the camera resource, but subclass 3 differs from subclass 1 in that in practice an exception occurs for - A delays getting the camera resource instead of failing to get it.

In the example demonstration, we use two common video calling and conferencing APPs, Fishu and WeChat, for illustration and demonstration. Firstly, open video call on WeChat and make it enter PIP mode, WeChat get focus, verify WeChat get camera resources, no exception; then open Fishu in the main interface for video conference, at this time Fishu get focus, verify Fishu get camera resources, WeChat lose camera resources, no exception; finally, let Fishu enter PIP mode, click WeChat small window to make it resume the main interface, at this time WeChat gain focus, verify that WeChat does not immediately get the camera resources, but waits a long time to get the camera resources, and the camera resources are still held by Feishu during this time when they are not obtained, and an exception occurs, that is, there is a delay in getting the corresponding camera resources after the focus switch.

1.2.4 Camera option automatically turns off

A and B need to use camera resources at the same time and B gets the focus, then make A become the main interface and find that the camera option in A which should be controlled by the user is turned off automatically, which is not what we expect.

In this example, we use two popular video calling and conferencing apps, Nail and WeChat, to illustrate and demonstrate. First, we open the video conference on Nail and put it into PIP mode, then we open WeChat in the main interface for video call, and finally we put WeChat into PIP mode and click on the small window of Nail to bring it back to the main interface, at this time the camera option in Nail Meeting is automatically turned off and an exception occurs, i.e. the camera option is turned off by a factor other than the user.

1.2.5 Typical APPs with the above exceptions in manual testing

APP	Subclass 1	Subclass 2	Subclass 3	Subclass 4
Wechat		√	√	
QQ	√	√		
Feishu	√	√		
DingDing		√		√
Tecent Meeting	√	√		
Small Fish Easylink	√	√		

1.3 Recording Resource Conflicts

Recording resource conflict exception refers to a situation where two applications, one in home screen mode and one in floating window mode, apply for system recording resources at the same time causing resource competition, thus causing one of the applications to be unable to correctly obtain resources and implement normal application logic. After we conducted a large-scale test on the application market app, we found that the recording resource conflict phenomenon existed in system phone, conference software, voice call software, instant messaging communication software and professional recording tool software.

The problems where the application logic could not be properly implemented due to the seizure of recording resources were classified into six sub-categories based on the observed situations.

- Cannot record and cannot send: When other applications occupy the audio focus, recording of voice strips cannot be performed and cannot be sent.
- Can send, but recording fails: When other applications occupy the audio focus and send a voice message, it appears that the message can be sent but the voice bar playback is empty.
- Call interruption: When the audio focus is taken away, the call is paused and automatically hangs up.
- Microphone off: The microphone is automatically turned off when the audio focus is captured and remains off when the call is returned.
- Volume reduced: When the audio focus is taken by another application and returned, the recording sound heard by the other party is reduced and does not return to normal volume.
- No sound: When one software uses audio, open another software to occupy the audio focus, at this time the software no longer radio, the other party can not hear the sound.

1.3.1 Cannot record and cannot send

For the example of not being able to record and not being able to send, we will demonstrate with Fishu Conference and QQ. First of all, we open Fishu Conference and make the application enter the small window mode, and the microphone is set to off, then we enter QQ and try to send the voice note, at this time, QQ application shows the permission prompt as "The microphone has no sound, maybe QQ's recording permission is prohibited. "This means that even if the meeting is in floating window application mode and the microphone is not turned on, it still occupies the focus of the recording device, which makes the QQ voice bar sending function not work properly and affects the normal use of the application.

Two application threads are competing for the recording resources at the same time, one application thread exists in the form of floating window and the other application thread exists in the form of cell phone main application, and due to the exclusivity of the recording device, it cannot serve more than one thread at the same time, so only one thread can take the focus of the recording resources normally and realize the recording function. In the actual application, if a user uses the conference software, even though he does not open the microphone, the recording resource is still seized, and if he opens WeChat in the form of a free window and tries to record and send a voice note, this operation will not be successful because the conference software does not release the recording focus at this time, which is contrary to the ideal user experience and the mobile operating system does not provide the service correctly. If an application captures recording resources in the form of a floating window at a certain moment, while the user is making an important call, it faces the risk of leaking recording files and capturing frequency information, jeopardizing the user's privacy.

1.3.2 Can send, but recording fails

For the example of sending, but the recording failed, to demonstrate the Fishu meeting with WeChat. In our test process, first enter the Fishu meeting, and turn off the microphone, set Fishu to floating window mode, then click into WeChat, try to send the voice bar, WeChat appears "Fishu is recording, so WeChat can't record" body reminder, although you can successfully send the voice bar, but the voice bar is empty, there is no There is no sound. It can be explained that when the Fishu application is in floating window mode and the microphone permission is not opened, it still captures the focus of the recording device, so that the WeChat application cannot realize the function normally.

1.3.3 Call interruption

For the call interruption example, we use WeChat and Fishu meeting application to demonstrate. First enter WeChat application, open WeChat call, then adjust WeChat call to small window mode, then click into Fishu and open Fishu meeting, audio focus is taken by Fishu, WeChat call connection is automatically disconnected.

1.3.4 Microphone off

For the microphone off example, we will demonstrate with FeiShu meeting and nailing meeting. Firstly, we open the nail meeting and set it to floating window mode, then enter FeiShu and open the FeiShu meeting, the microphone of the nail application is originally in the open state, and it will be closed automatically after the focus of the device is taken by FeiShu.

1.3.5 Volume is reduced

For the example of volume reduction, we will demonstrate with Fishu meeting and WeChat voice call. First click into Fishu to open Fishu meeting, and set it to floating window mode, then click into WeChat call, at this time, the audio focus is taken by WeChat, and the audio volume in Fishu meeting is reduced.

1.3.6 No sound

For the example of no sound, the system phone, fly book meeting to demonstrate. First click into the fly book to open the fly book meeting, and set it to small window mode, then open the system phone, the other end of the phone feedback for "no sound", that is, in the case of fly book to capture the focus of the recording device, the system phone can not correctly get the focus, affecting the normal use.

1.3.7 Typical APPs with the above exceptions in manual testing

APP	Subclass 1	Subclass 2	Subclass 3	Subclass 4	Subclass 5	Subclass 6
Wechat		√	√			√
Feishu					√	√
Tecent Meeting					√	√
DingDing				√		
Jinshan Meeting						
Wangyi Meeting						√
National karaoke	√					
Xunfei Input	√					

Method						
Tencent Classroom						√
System Phone						√
QQ	√					√
System recording	√					

1.4 Video Playback Conflict

Video playback conflict exception refers to a situation where two applications, one in main application mode and the other in floating window mode, can play video normally in one application while the other application cannot implement the application logic normally. We launched a large-scale test of the application market APP and found that most video playback software did not handle this type of exception.

The video playback conflicts were divided into five subcategories based on the observations.

- Small window play pause: open a video software to play video in PIP mode, at this time, if you click another video software for PIP video playback, the previous video playback is suspended.
- Small window disappears: open a video software to play video in PIP mode, if you click another video software for PIP video playback, the previous video playback PIP window disappears.
- Unable to open the small window: No response after clicking the small window button.
- Cannot open PIP mode, only background sound: the software may not have developed PIP mode, the video can only be played in the background in the form of sound, and cannot be displayed in picture-in-picture mode.
- PIP mode is not developed: The video software does not have a button to enter picture-in-picture mode, and cannot set the small window, and has not developed this function.

1.4.1 Floating window play pause

For the small window play pause, in our test software, Mango TV, Tencent video picture-in-picture conflict as an example for demonstration. First of all, enter MANGO TV to open the video, and then adjust to the small window mode, and then enter Tencent video, click into the video playback, at this time, the small window video, the main window video can be played normally, if at this time click Tencent video small window button, so that the same into the small window mode, MANGO TV video small window playback pause, Tencent video small window normal playback. The two appear video focus on resource competition conflict.

1.4.2 Floating windows disappear

For the disappearance of the small window, in our test software, the Youku video, Mango TV picture-in-picture mode conflict as an example for demonstration. First enter Youku video and click to open a video, then adjust to picture-in-picture mode, then enter Mango TV, also click to enter the video playback, at this time click the small window button, and observe that the small window of Mango TV appears successfully, but the small window of Youku video disappears.

1.4.3 Cannot open floating window

For the inability to open the small window, in our test software, Tencent video, Youku video picture-in-picture mode conflict as an example to demonstrate. First enter Youku video, click to enter the small window playback mode, then open Tencent video, click the small window button does not respond, the video located in the main screen can not normally enter the small window mode.

1.4.4 Unable to open PIP mode

In response to the inability to open PIP mode, in our test software, good-looking video appears the exception. That is, click into the home screen video playback, at this time click the home screen button, the application transferred to the background running, at this time only the background sound playback, but in the screen did not have a small video window screen appears.

1.4.5 No PIP model developed

For applications that have not developed PIP mode, we still classify it as an exception, and this exception occurs for Dodo Video in the software we tested.

In the case of video software in undeveloped floating window mode, the logic of using the video focus does not have the problem of dichotomy. According to the normal implementation logic, the software under the main application gets the video focus and plays the video normally. However, in the floating window (either free window or picture-in-picture window) mode, the use of video focus by two videos may conflict, and the operating system does not handle the relationship between the main application and the floating window application threads well. The other application can only be forced to pause playback, or only have audio in use, or even close the floating window directly. During the actual test, we found that only a very small number of video playback software handled the threaded state of the video in the main application and picture-in-picture mode, and the very majority of video applications would still have video focus competition conflicts with the main application even in floating window mode.

1.4.6 Typical APPs with the above exceptions in manual testing

APP	Subclass 1	Subclass 2	Subclass 3	Subclass 4	Subclass 5
Youku Video	√	√			
Aiqiyi Video	√	√			
Tencent Video	√		√		
Sohu Video	√				
Bilibili	√				
Watermelon Video			√		
Douyu TV	√				
Mango TV					

Good video				√	
Toto Video					√

1.5 App Black Screen Exception when Recording Screen

App black screen abnormality when recording screen refers to the situation where the application is opened in free window or picture-in-picture mode with the system recording screen turned on by the phone, and the application is unprotected or over-protected due to its own design defects. Under the application that does not use floating window, the application recording screen involving privacy is protected by black screen, and the application that does not involve privacy is allowed to capture the application screen, which is in line with the logic of application protection and recording screen function use. However, some applications do not take into account the security issues under the floating window, so that the recording screen is feasible under the floating window, which greatly increases the security risks; some applications do not have special treatment for the black screen protection under the floating window of the application, and choose the full-screen black screen mode, which affects the normal use of the recording screen function by users; some applications lack completeness in the black screen processing of the application recording screen under the floating window, such as only the password input For example, only the password input, ID card input part of the black screen protection, but the verification code input, face recognition and other parts of the black screen protection is not, which poses a threat to the user's personal privacy.

According to the observed situation, APP black screen abnormalities during screen recording are divided into 3 sub-categories.

- Applications involving privacy are not protected by black screen: the content of the application involves user privacy, but the application designer only considers the privacy protection of the application in the main screen mode, and the internal operation of the application in the floating window mode can still be recorded by the recording screen.
- The black screen range is too large: the application has privacy protection processing, but the black screen range is too large, beyond the size of the floating window, for the floating window mode APP, ideally it should be black inside the floating window, but the actual will be full screen black screen, affecting the user use.
- Incomplete black screen protection: only when the Huawei security keyboard appears will the black screen, if the security keyboard is retracted then the password already entered can be seen.

1.5.1 Applications involving privacy are not protected by black screen processing

For the case of no screen recording exception protection, there are Taobao, WeChat and Alipay in our test software with such exceptions. For Taobao, it requires password input when confirming the receipt of goods, but the application does not protect the black screen at this time, and after observing the recorded file, we found that the entire process of password input by the user can still be observed in this part; for WeChat, when entering the payment interface, it does not black screen the graphic lock input, which exposes the user's WeChat graphic lock password to the risk of leakage; for

Alipay, the graphic lock when entering the payment For Alipay, the graphical lock in the payment interface is still unprocessed, which also has the risk of leaking the graphical lock password.

1.5.2 Black screen range is too large

For the black screen range is too large, most of the software tested showed the feature, that is, cover the screen range is too large, resulting in abnormal use of the recording screen function. Such as the use of floating window mode of Jingdong payment, cloud flash payment and WeChat payment, etc., will be black screen on the entire screen, in the floating window mode of Taobao when entering the login password, will still be black screen on the entire screen.

1.5.3 Incomplete black screen protection

For incomplete black screen protection, such anomalies occurred in our test software Huawei Secure Keyboard. When we entered the login password to an application in floating window mode, the screen would only go black when Huawei Secure Keyboard appeared, but after retracting Huawei Secure Keyboard, the password already entered could still be seen.

1.5.4 Typical APPs with the above exceptions in manual testing

APP	Subclass 1	Subclass 2	Subclass 3
Taobao	√		
Wechat	√	√	
Alipay	√		
JIngdong			
Spellbound			
Meituan			
Party Building Cloud		√	√
Idle fish			
Vipshop	√		
Bank of Communications			
Construction Bank			
China Merchants Bank			
QQ		√	
SuperStar Learning Pass		√	
VolunteerHub		√	
Know to		√	
Learning to be strong		√	
Tianjin Human Resources and Social Security		√	
Tsinshin Office		√	
Tianjin Metro		√	
QQ Email		√	
Zheng Hao Office		√	

1.6 Keyboard Problems

The keyboard class problem refers to the anomaly that occurs when an APP is opened in FreeForm (free floating window) mode as A and an APP is opened in the main interface as B. When both A and B can invoke the keyboard at the same time, the keyboard of both APPs is invoked in turn by switching between the two APPs with a click.

To facilitate the narrative, we define several concepts that appear in this section.

1. system keyboard: the keyboard pops up for users to use when they do text input on the phone, and we call it the system keyboard for the most common input keyboard provided by the system. For example, the keyboard used for SMS input, memo input, chat software input, etc. 2.

2. System security keyboard: When entering passwords and authentication codes involving security and privacy on the phone, a security keyboard will pop up for the user to use, for this kind of security input keyboard provided by the system, we call it the system security keyboard. For example, the Huawei secure keyboard provided in Huawei cell phones is called by Learning Strong, 12306, Tianjin Metro and other APPs when entering passwords.

3. application built-in keyboard: in some applications with high security requirements, the APP itself provides a special keyboard for users to enter passwords or verification code input, we call it the application built-in keyboard. For example, Alipay payment, WeChat transfer/payment, China Construction Bank, China Bank of Communications, etc. all provide in-app keyboards.

The keyboard type issues are classified into 2 sub-categories based on the observations.

- Wrong type of keyboard being popped up
- Multiple keyboards exist at the same time causing occlusion

1.6.1 The wrong type of keyboard is ejected

In FreeForm (free floating window) mode, we open an APP as A and an APP as B in the main interface. When A and B can call the keyboard at the same time, we take turns to call the keyboard by clicking the input box in A and B. We find that an APP pops up the keyboard which is not used under normal circumstances.

In the example demonstration, we use two common password/text input apps, Memo and China Construction Bank, for illustration and demonstration. Firstly, open Memo and China Construction Bank in the main interface respectively, click the input box to invoke the keyboard, verify that both of them pop up the system keyboard and in-app keyboard respectively; then open Memo in FF mode and China Construction Bank in the main interface, take turns to invoke the keyboard of both apps (first click to invoke Memo keyboard, then click the input box of China Construction Bank to invoke the keyboard of Construction Bank), the former has no abnormality. And the latter pops up the system security keyboard instead of the application built-in keyboard we expect, an exception occurs, i.e. the wrong type of keyboard is popped up.

1.6.2 The simultaneous presence of multiple keyboards causes obscuration

Similar to subclass 1, but when clicking in turn, the exception occurs when the keyboards of both apps appear on the screen, instead of keyboard A pops up when keyboard B automatically closes, resulting in the keyboard obscuring the screen information and the user cannot see other content on the screen, creating a security risk and bringing a poor user experience.

In the demonstration, we use 12306 and Bank of Communications, two common password/text input applications, to illustrate and demonstrate. First, open the login password input interface of 12306 and Bank of Communications respectively in the main interface, click on the input box to invoke the keyboard, and verify that the system security keyboard and the application built-in keyboard pop up respectively; then open 12306 in FF mode and Bank of Communications in the main interface, and invoke the keyboards of the two applications in turn (first click to invoke the keyboard of Bank of Communications, and then click to invoke the keyboard of 12306). It was found that the former pop-up application built-in keyboard and the latter pop-up security keyboard appeared in the screen at the same time and blocked most of the information in the screen, and an exception occurred, i.e. multiple keyboards existed at the same time causing blockage.

1.6.3 Typical APPs with the above exceptions in manual testing

APP	Subclass 1	Subclass 2
SuperStar Learning Pass	√	
12306		√
Bank of Communications		√
China Construction Bank	√	
Wechat	√	

2 Automation Testing

In the previous section, we categorized and summarized the floating window exception types by manual testing, however, manual testing is not feasible due to cost constraints when facing the testing of high-volume applications at market scale. Based on this, we propose an automated testing solution to write automation test scripts for six types of exceptions, laying the foundation for further expanding the testing scale and improving applicability and pervasiveness.

In this section, the automation test program is first designed and implemented one by one based on the functional requirements in the design. For the overall implementation tool of the automation test program, the API provided by the UIautomator2 test framework is used for program writing, and combined with the weditor tool for the identification and positioning of controls. Since the settings and distribution of controls vary greatly from one application to another, we started with small batch automation test program writing for different functional features of applications, and on this basis, we analyzed the program and tried to find common operations for automation testing of applications in the market.

2.1 Automated testing of penetration phenomena

Design process:

Penetration occurs when one application is opened in the home screen and another is opened in PIP or FF mode, and successive clicks on a component of the upper application cause that component to respond to a component of the lower application in the corresponding location on the screen. In the testing of this phenomenon, the automated operations we performed included opening the two test applications in home screen and floating window mode, clicking on a coordinate of the upper application at regular intervals, and observing whether the lower application was clicked on due to the penetration phenomenon. During the whole process, it is necessary to ensure that there should not be any interfering clicks and touch events on the window under test except for a certain interval of clicks on a certain coordinate, so as to avoid the response of the lower application due to other factors.

The runtime flow is mainly as follows

- Click on the home screen to open application A
- Open application B in FF mode via the sidebar
- Click on a coordinate in Application B continuously at 1.0 second intervals

Program script:

We write automation scripts to implement the above operation flow. For the sidebar opening, the operation to be completed is to slide inward from the edge of the screen and stay for a certain period of time. Since swip, drag and other methods cannot solve the operation problem well, we use the method of small-amplitude move until the time reaches enough to make the sidebar open and then complete the click; for the sidebar opening of the application, use the relative positioning For the opening of the application in the sidebar, we use the relative positioning method to find the clickable control above the name for the click operation; for the continuous click, we use the click_gone method and set the interval to 1.0.

In this automated test, the application under test is application B, which is opened in floating window mode, and application A, which is opened on the main screen, can be chosen freely, but it is necessary to ensure that when the penetration exception occurs, application A will respond and the response can be easily observed. At the same time, we need to ensure that the location of the click is the location of the application B that does not generate a response after the click, to avoid the error caused by the continuous click that causes the application B to keep jumping around the page.

In summary, we need to choose the location of the persistent click to ensure that the click coordinates in the upper application can not generate a response and in the lower application can generate an easily observable response.

Experimental procedure:

We conducted an experiment using the above automation script to verify the correctness of the automation test penetration problem. After the test device is connected, we can see that QQ is opened on the main screen, the sidebar is opened by left swipe, and WeChat is opened in FF mode, and the "clickable" option in WeChat is clicked every second.

2.2 Automated testing of camera/recording resource conflicts

Design process:

Camera call conflicts occur when one application is open as a home screen and another application is open in PIP or FF mode, and when both need camera resources at the same time, the application that gets the focus is unable to get the camera resources (in time). The recording resource conflict occurs when one application is opened as a home screen and another application is opened in PIP or FF mode, and when both need recording resources at the same time, one application cannot get the resources properly to implement the application logic. In the testing of this phenomenon, it becomes complex and unnecessary to complete automation scripting for the whole process because there are interactions generated with constant switching of dialing, answering, PIP and FF window modes, and there are multiple possible situations and sequences such as meetings and video calls. Therefore, we propose the method of dividing the entire test process into several "atomic operations", which can be combined to obtain automation scripts for various test situations in the actual test.

The specific "atomic operations" are as follows:

"Atomic operations"
Video call dialing
Video call answering
Opening of the video conference
Video conferencing added
Open a video call/conference in FF mode
Open a video call/conference in PIP mode
Recorded voice bar

When testing, these "atomic operations" need to be combined to obtain a complete test flow.

Program script:

We write automation scripts to implement the above running process. For the video conference opening and joining operation, complete.

- Find the video conference portal and enter
- Create a new conference and enter/enter the conference number to join the conference
- Turn on the microphone and camera of the device

For making and receiving video calls, complete.

- Find the video call function in the "More" option
- Click to make a video call
- Click on the answer button to answer the video call

For opening a video call/conference in FF/PIP mode, complete.

- Open the application in FF mode via the sidebar
- Two ways to enter PIP mode

The way to open in FF mode through the sidebar is the same as the one in 2.1. Meanwhile, for different applications, the methods to enter PIP mode are different. The two most common methods to enter PIP mode are: (1) directly exit the application to run in the background, the application will automatically enter PIP mode. First, find the bottom edge of the screen by window_size method, and use the swipe method to finish swiping up to launch to the background. (2) Click the special small window function inside the application to enter PIP mode. Common applications that belong to the former are: Fishu Conference, QQ video call, WeChat video call, etc., and those that belong to the latter are: Tencent Conference, etc.

For the operation of recording voice bars, complete.

- Click into one of the chat windows

- Find the recording bar option
- Long press the record button for voice recording and sending

Experimental procedure:

We used the above "atomic operation" automation script combination and conducted experiments to verify the correctness of the automation test camera/recording resource conflict problem.

For two devices A and B, the automation script on device A consists of: initiating a video conference, waiting, and making a video call; the automation script on device B consists of: joining a conference, entering PIP mode, answering a video call, entering PIP mode, clicking back to the conference, and repeating the above three steps. After running the combined automation test script, it is observed that the camera resources on device B are always held by the video conference, and when the focus switches to the video call, the video call still does not get the camera resources.

To test the problem that the voice bar cannot be sent in the recording resource conflict, for example, for two devices A and B, the automation script on device A consists of: initiating a meeting, waiting; the automation script on device B consists of: joining a meeting, meeting into PIP mode, and recording a voice bar. After running the combined automation test script, it is observed that the voice bar cannot be sent out successfully on device B and an error message pops up.

2.3 Automated testing of video playback conflicts

Design process:

Usually, video playback conflicts occur when a video is opened in the form of a main window and another window is opened in picture-in-picture mode, or both windows are opened in picture-in-picture mode. In the above-mentioned cases, the audio and video focus (also known as audio and video resources) of the host is captured by one video application, thus causing the other video application to play abnormally and appear to pause or disappear. Based on this, we design an automated test program to implement video playback conflict testing between different video software. This automated test program should contain the following functions.

- Ability to open a specified video application.
- Click to enter a video application and tap on the video to play it in home screen mode.
- Click on the video being played to enter picture-in-picture mode.
- Test two or more video applications at the same time, i.e. be able to put one or more video applications in picture-in-picture mode and the remaining video applications in main application mode.
- Add automatic screen recording function, which can record the testing process of video playback conflicts for later observation and analysis.

Program script:

The process focuses on the implementation of the features mentioned in the automation program design one by one. The same UIautomator2 automation testing framework was used to write the test, combined with the Weditor tool for element identification and location. Starting from small batch video automation test program writing, we try to find common operations for automation testing of video applications in the market.

For opening the specified video application, we use the device.app_start() interface provided by UIautomator2 to implement it. In the preliminary preparation, we count the package names of different video applications and record them; for; for clicking the video for main screen playback, we first scan the page of the video application to find the specified For the entry of video picture-in-picture playback mode, we researched a large number of video applications and divided them into two categories, one for manually clicking into picture-in-picture and one for automatically entering picture-in-picture by returning to the main screen, the former by finding-click mode and the latter by returning to the main screen operation; for two or more For testing of two or more video applications, the first three functions are realized through a combination of the first three functions, written through the logical sequence of the program of clicking on the application-entering video playback-entering picture-in-picture mode; for the screen recording function, it is realized through a specific screen coordinate click, device(description='screen recording') click.

Experimental procedure:

We conducted experiments to evaluate the above automated testing procedures. The results show that the program is able to open video applications, play video on the main screen, play video in picture mode, test for video conflicts, and automatically record screens.

2.4 Automated testing of APP black screen exception when recording screen

Design process:

Usually, the black screen exception of APP when recording screen happens when the phone opens system recording screen or conference software opens recording screen, the application opens in picture-in-picture mode or free window mode, and the problem that the content of sensitive application window is not protected, and the content of application window is over-protected (other than the window is black). In the above case, the application writer does not handle the state of the application in floating window separately, resulting in the problem of abnormal black screen of the APP when recording the screen. Based on this, we design the automation test program to implement the test of floating window exception when recording screen for different software. The automated test program should contain the following functions.

- Be able to open the system recording screen.
- The ability to click into an application.
- The ability to enter the application's sensitive information input or display page (such as entering password, user information display, etc.).
- The ability to put the application into free window mode or picture-in-picture mode.

Program script:

The process focuses on the implementation of the functions mentioned in the automation programming one by one. It is also implemented using the UIautomator2 framework and is based on Weditor for element identification and positioning. We tested some of the applications for black screen anomalies during screen recording and tried to find common operations for testing such anomalies in the market applications.

For opening the system recording screen, first open the notification bar through device.open_notification(), then find the start button through device(description='screen recording') to record; for clicking into a certain application, take pre-storing the application name and application package name information to open; for opening the sensitive information page Need to determine the distribution of password input and user login elements through manual observation in advance, and manually determine the sensitive information, and then realize it in the subsequent automation program writing process; for the application to enter the free window or picture-in-picture mode, it is realized through the floating window function provided by the operating system, the process is main screen - application background situation - menu bar in the upper right corner - enter the floating window mode.

Experimental procedure:

We conducted experiments to evaluate the above automated testing procedures. In this section, we tested these applications: Fishu, QQ, WeChat, Tencent Video, Taobao, Genco Sports, YouTube, TikTok, Twitter, Teams, Netflix, Zoom. The above applications were collected from domestic APP application market and overseas Google application market. The experimental results show that the automated test program can achieve the functions of recording screen opening, application opening, entering the sensitive information display page of the application, and making the application enter picture-in-picture mode or floating window mode.

2.5 Automated testing of keyboard-like problems

Design process:

The keyboard class problem occurs when one application is opened in the form of home screen and the other application is opened in FF mode, and the exception occurs when switching the keyboard between the two. In the test of this phenomenon, we divide the applications that need to call the keyboard into social communication category and security account password category, where the social communication category includes QQ, WeChat and other applications, the keyboard call occurs when the text is entered in the chat box; the security account password category includes Bank of Communications, China Construction Bank, 12306 and other applications, the keyboard call occurs when the transfer password or login password is entered.

The operation flow is mainly as follows.

- Click to open application A on the home screen
- Open application B in FF mode via the sidebar
- Both applications find a screen with an input box
- Click on the input box of A and B respectively to complete the keyboard pop-up

Program script:

We write automation scripts to implement the above runtime. For the implementation of the sidebar opening, the same method as in 2.1 is not repeated; for the implementation of the keyboard pop-up by clicking the input box, the input box needs to be identified by the class of the control. We have investigated the input box class (className) of some common applications in the market as follows, and summarized them as (1) android.widget.ScrollView (2) android.widget.EditText. Since some applications with strong security such as Bank of Communications and China Construction Bank encrypt their controls, it is impossible to view their types accurately, so the keyword identification method is used to find the location of the corresponding control for this case.

Common Applications	Input box control type
Wechat	android.widget.ScrollView
QQ	android.widget.EditText
Feishu	android.widget.ScrollView
SuperStar Learning Pass	android.widget.EditText
Railroad 12306	android.widget.EditText
Tianjin Metro	android.widget.EditText
Bank of Communications	\
China Construction Bank	\
Learning to be strong	android.widget.EditText
VolunteerHub	android.widget.EditText
Know to	android.widget.EditText

In social communication applications, you need to click on any chat window to find the location of the input box and pop up the keyboard by identifying and judging the category of controls; in secure account password applications, if it is a transfer password, you need to jump to the transfer interface and find the corresponding control and pop up the keyboard based on the keywords "input", "password", etc. "If it is a login password, you need to jump to the login screen and find the corresponding control and pop up the keyboard according to the keywords of "input", "password", etc. or according to the category of input box. If it is a login password, you need to jump to the login screen and find the corresponding control and pop up the keyboard based on keywords such as "enter", "password", etc. or based on the type of input box.

The method of finding the login page in different security account password applications has some commonality but also has great differences. Since it is too redundant and of little value to consider the extremely individual cases into the scripting, we use a general approach with small "personalization" to find the most common cases We summarize and script the most common cases, and give special consideration to special cases.

Experimental procedure:

We use the above automation script to conduct experiments to verify the correctness of the automation test for keyboard-type problems. Take the application opened on the main screen is WeChat and the application opened in FF mode is China Construction Bank as an example, run the automation test script, you can see that after the test device is connected, open WeChat on the main screen and enter a certain chat window, left swipe to open the sidebar and then open China Construction Bank in FF mode and then gradually click to enter the transfer login interface, click the input box of the two applications respectively to bring up the keyboard. It was observed that the keyboard popped up by WeChat caused obscuration to the keyboard popped up by

China Construction Bank.