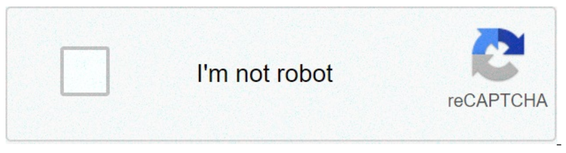

Continue

Alfred ip camera app

Alfred is described as "app that turns your smartphones to home security cameras and allows you to be able to know wherever you are. The features of motion detection and push notification warn when something is happening '. There are more of 10 alternatives to Alfred for a variety of platforms, including Android, iPhone, iPad, Android and Mac tablets. The best alternative is Athome room, which is free. Other fantastic apps like Alfred are Athome Video Streamer (free), Cawice (Freemium), why (free) and everything (Freemium). Posts from the Alfred Camera engineering team, note that the information, uses and applications expressed in the post below are only those of our guest author, camera Alfred. In this article, D WEA as to give you a brief overview of Alfred Camera and our experience of use MediaPipe to transform our object function into motion, and how MediaPipe has contribute To get the easiest things to achieve our goals. What is the Alfred camera? Fig.1 Alfred Camera Logo Alfred Camera is an intelligent home application for Android and iOS devices, with over 15 million downloads around the world. By downloading the app, users are able to transform their backup phones into security cameras and monitors directly, which allows them to look at their homes, shops, pets at any time. The Alfred camera mission is to provide domestic security at affordable prices so that everyone can find tranquility in this busy world. The Alfred Camera team consists of professionals in various fields, including an engineering team with various automatic learning experts and computer vision. Our goal is to integrate technology to in devices accessible to all. The learning of the Alfred Camera Alfred Camera machine currently has a function called motion object detection, which continuously uses the device's camera to monitor a destination scene. Once you identify a moving object in the area, the app will start recording the video and send notification to the device owner. Automatic learning models for detection are and trained by our team using TensorFlow, and run on the Tensorflow Lite with good performance even on medium-handed mid-range devices. This is important because the app is taking advantage of the old phones and we would like the function to reach as many users as possible. The challenges we had begun to build ours at the main features Alfred Camera since 2017. In order to have a solid base to support our needs of AI for the coming years, we decided to rebuild our real-time video pipeline analysis. At the beginning of the project, the goals had to create a new pipeline that should be 1) quite modular so you can easily exploit the main algorithms with minimal changes in other parts of the pipeline, 2) with GPU acceleration designed in position, 3) Cross Platform as much as possible, so you don't need to create / maintain separate implementations for different platforms. Based on the objectives, we had interviewed several open source projects that had the potential, but we ended up having to use any of them because it fell short on the features or we were not providing the availability / the stories we were looking for. We started a small prototype team on those goals first for the Android platform. What has arrived later there were some difficult challenges above what we originally expected. We came across several important design changes since some key design bases were We needed to implement some utilities to do things that sounded trivial but they requested a significant effort to make it right and fast. Treating with asynchronous processing also took us to a pile of timing problems, which took the team a bit of effort for the address. Not to mention the debugging on real devices was extremely inefficient and painful. Things didn't stop here. Our product is also on iOS and we have to face these challenges again. Furthermore, the discrepancies in the behavior between the specific implementations of the platform have introduced further problems we had to solve. Even if we finally managed to get the IL At the confidence level we wanted, which was not a very pleasant experience and we never stopped thinking if there is a better solution. MediaPipe - a Changer Google Open Project MediaPipe game from June 2019 and is immediately attracted to our attention. We were surprised by the way it is perfectly aligned with the objectives we have set precedents, and has functionalities that could not have been developed with the amount of engineering resources we had as a small company. We have immediately decided to start an evaluation project with the construction of a new product feature directly using MediaPipe to see if it could live up to all the promises. Migration to MediaPipe To start evaluation, we decided to migrate our existing motion object function to see what exactly MediaPipe can do. Our current survey pipeline object is constituted by the following main components: (Moving) Object detection model as explained above, a Tensorflow Lite model trained by our team, tailored to operate on mid-range devices. Scarce detection lighting and low light filter Calculate the average scene luminance, and based on the result conditionally process the input frames to intensify the brightness of the pixels to allow our users see things in the dark. We are also checking whether or not to be performed as the motion object detection model does not work properly when the frame was processed by the filter. Motion Detection Sending Frame Through Moving Object Detection still consumes a considerable amount of energy even with a small model like what we have created. Inferences always in operation does not seem to be a good idea, as most of the time you can't be any object moving in front of the camera. We decided to implement a gating mechanism, where the frames are only sent to the object of the motion detection model based on the movements detected by the scene. The detection takes place mainly by calculating the differences between two frames with some additional tricks that take the movements detected in a few frames first into consideration. Area of interest This is a mechanism to allow users to manually mask the area where the camera does not want to see. It can also be performed automatically on the basis of regional luminance which can be generated by the aforementioned detection component of poor lighting. Our current implementation has considered as GPU as much as possible. A series of shaders are created to carry out the tasks mentioned above and the pipeline was designed to avoid moving pixels between CPU / GPU frequently to eliminate potential performance shots. The pipeline involves multiple ML models that are performed in a conditional, mixed CPU / GPU processing, etc. All the challenges here make it a perfect showcase for what MediaPipe could help develop a complicated pipeline. Playing with MediaPipe MediaPipe provides a lot of code examples for any bootstrap developer with. We took the survey object on the Android sample that comes with the project to start due to the similarity with the back-end part of our pipeline. He did it sometimes take us to fully understand the design concepts of MediaPipe and all the associated tools. But with the complete documentation and the great response capacity by the MediaPipe team, we arrived â €