GitHub Username: https://github.com/jimduan96/

Vehicle Driver e-License

Description

This app using smartphone front view camera instant capture the driver's facial image; processing the image comparison algorithm with authorized users' facial photos, to achieve the purpose of identification of the driver. And the phone communicates with Android Auto equipped Head Unit to obtain vehicle information. Driver face information, phone App information, and vehicle information are binded together to send along to the Cloud server to process security authorization or license confirmation. By returning successful results, the car started to play some predefined functions, such as starting the vehicle, speed limits, regional traffic and so on.

The App scalable refinement functions and the use of electronic vehicle key, e-Driver's License, intelligent transportation, car rental, etc.

Intended User

For automobile driver who drives a vehicle which has Android Auto head unit installed.

(Others: Fleet vehicle management, Smart car component...)

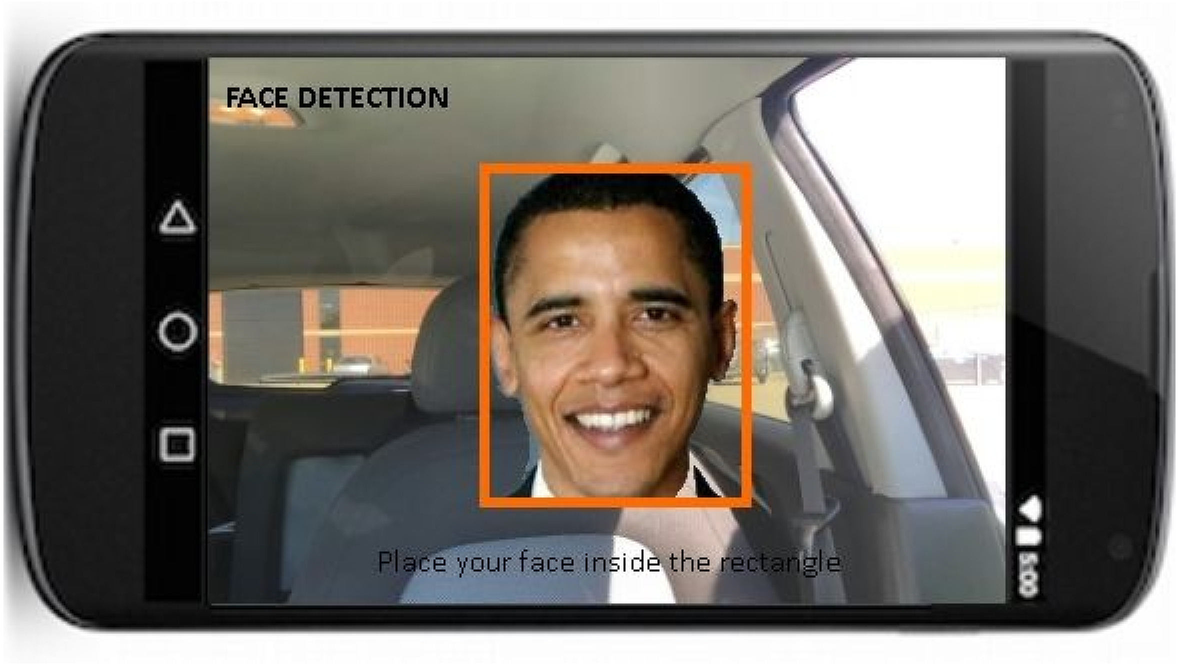
Features

Main features:

- Instance collect driver facial image
- Machine recognize human faces
- Support Multi-user
- Replace physical vehicle keys
- e License
- communicate with Android Auto
- Cloud server authentication centralized

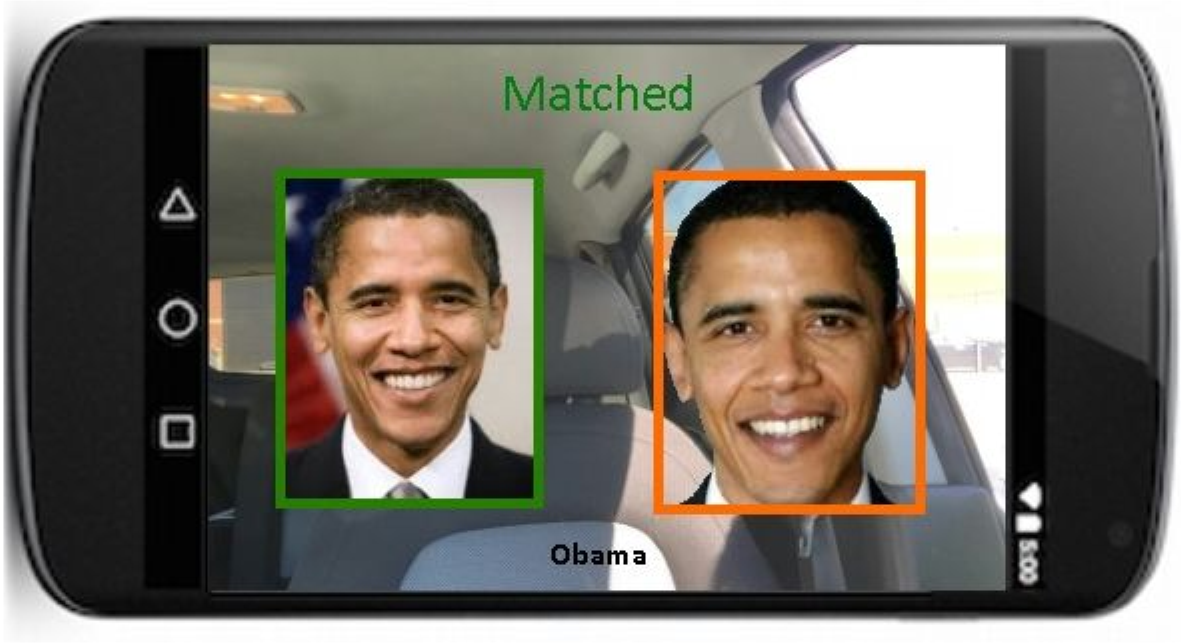
User Interface Mocks

Screen 1



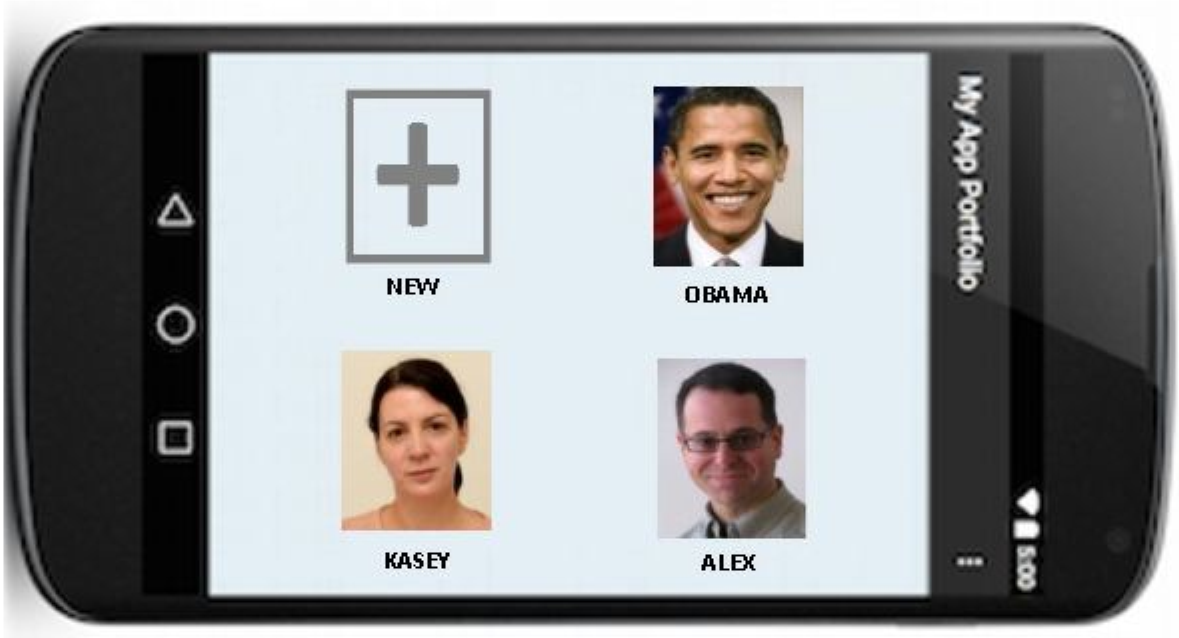
The front camera view shows up each time App launched. App will auto detect face image after driver face fitted into the orange rectangle. And the rectangle color orange will be changed to green color if face has been detected.

Screen 2



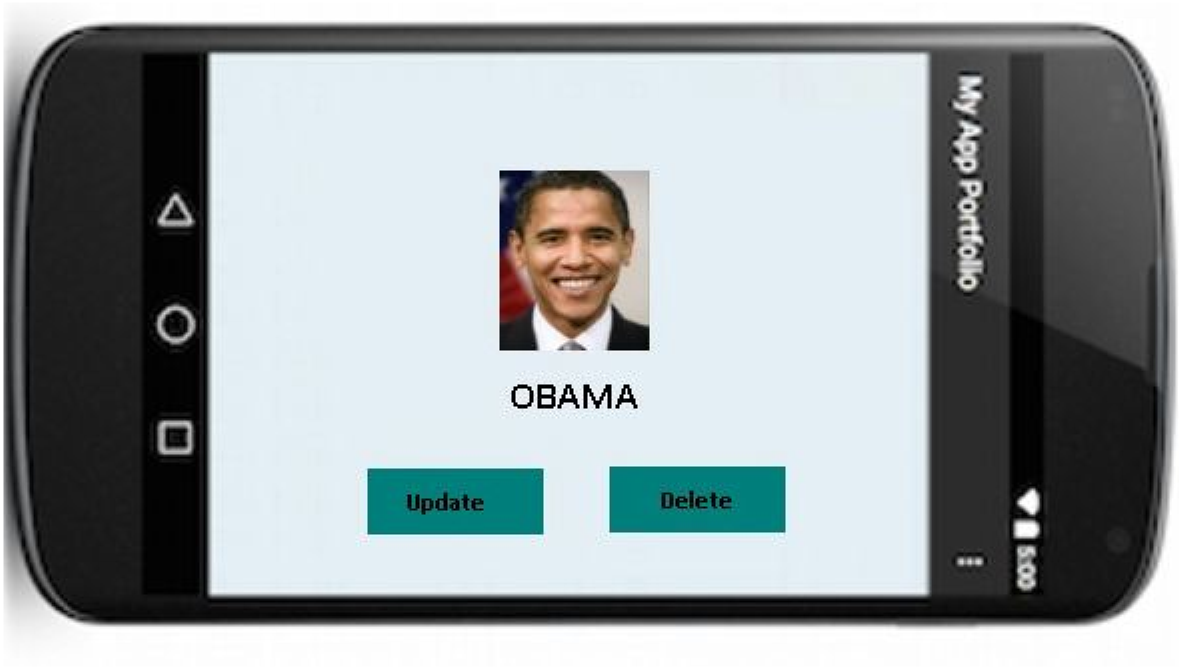
App does the face recognition algorithm after face has been captured. Above view shows matched both images of pre saved and recent captured.
At the background, phone retrieves a “car key” from Android Auto equipped vehicle head unit. Phone takes matched face id combined with “car key” to transmit to Cloud to finish the authentication process.
Phone passed the authentication result to vehicle to enable certain features predefined.

Screen 3



This view will be launched by menu item “User Configuration” clicked. This feature should be password protected. All saved user face images will be listed with name except the first spot reserved to adding new image.
Long press on any face image: “Screen 4” will pop up.
Long press “new (+)”: “Screen 1” will show up to allowing user capture new face image.

Screen 4



This is the screen to update or delete existing face image.
Button “Update”: the “Screen 1” to show.
Button “Delete”: the “Screen 3” will be shown with updated list.

Key Considerations

How will your app handle data persistence?

App builds a local file database to hold photo files.

Describe any corner cases in the UX.

After back button clicked on each screen, screen will switch to:
“Screen 1” -> System home screen
“Screen 2”-> “Screen 1”
“Screen 3” -> “Screen 1”
“Screen 4”-> “Screen 2”

Describe any libraries you’ll be using and share your reasoning for including them.

Opencv4Android: image processing for face detection and recognition.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure OpenCV library
- Download OpenCV SDK
- Port to Android Studio
- Setup Android Auto Simulator
- Launch AVD – Android Auto Desktop Emulator

Task 2: Implement UI for Each Activity and Fragment

- Create front camera viewer and process photo capture
- Enable front camera feature with capture function
- Photo storage
- Design all UIs
- Draw green/orange rectangle on camera view
- Menu
- List all photos

Task 3: Your Next Task

- Design Seed & Key algorithm
- A simple authentication (simulation)
- Implement Google Cloud service
- Pass "phone/face ID" and "vehicle ID" to Cloud; and return the result by above Seed & Key algorithm

Task 4: Your Next Task

- Face Detection
- Face Recognition

Task 5: Your Next Task

- Create Phone – Car – Cloud Communication
- Handle Error Cases
- Release App build