DOI: https://doi.org/10.36910/6775-2524-0560-2021-42-20
УДК: 004.38, 004.42
Костючко Сергій Миколайович, к.т.н., доцент
https://orcid.org/0000-0002-1262-6268
Багнюк Наталія Володимирівна, к.т.н., доцент
https://orcid.org/0000-0002-7120-5455
Кузьмич Олена Іванівна, к.ф.-м.н., доцент
https://orcid.org/0000-0002-8717-4497
Поліщук Микола Миколайович, к.т.н.
https://orcid.org/0000-0002-1218-5925
Кирилюк Людмила Миколаївна, асистент
https://orcid.org/0000-0002-8279-3133
Луцький національний технічний університет

БІОМЕТРИЧНА ІДЕНТИФІКАЦІЯ ЗАСОБАМИ PYTHON ТА RASPBERRY PI

Костючко С., Багнюк Н., Кузьмич О., Поліщук М., Кирилюк Л. Біометрична ідентифікація засобами Python та Raspberry Pi. У даній статті пропонується один з варіантів біометричної ідентифікації, а саме розпізнавання обличь користувачів засобами (одного з найпопулярніших і сучасного) одноплатного комп'ютера Raspberry Pi 4b та з використанням мови програмування Python. Була проведена низка симуляцій та створена база даних користувачів.
Ключові слова: Python, Raspberry Pi, FaceID, біометрія, OpenCV.

Костючко С., Багнюк Н., Кузьмич А., Полищук Н., Кирилюк Л. Биометрическая идентификация средствами Python и Raspberry Pi. В данной статье предлагается один из вариантов биометрической идентификации, а именно распознавания лиц пользователей средствами (одного из самых популярных и современного) одноплатного компьютера Raspberry Pi 4b и с использованием языка программирования Python. Был проведен ряд симуляций и создана база данных пользователей.
Ключевые слова: Python, Raspberry Pi, FaceID, биометрия, OpenCV.

Kostiuchko S., Bahniuk N., Kuzmych O., Polishchuk M., Kyryliuk L. Biometric identification by means of Python and Raspberry Pi. This article proposes one of the options for biometric identification, namely face recognition by means of (one of the most popular and modern) Raspberry Pi 4b single-board computer and using the Python programming language. A number of simulations were carried out and a user database was created.
Keywords: Python, Raspberry Pi, FaceID, biometrics, OpenCV.

Вступ

При розблокуванні телефону (FaceID) або дозволяєте Google або Apple сортувати ваші фотографії, ви використовуєте програмне забезпечення для розпізнавання обличь. Багато ПК з Windows також дозволяють використовувати ваше обличчя для входу. Цю процедуру (біометричної ідентифікації) можна використовувати не тільки для входу, засобами Raspberry Pi можна використати дану процедуру і для більш цікавих речей, а також для навчання та розвитку вже існуючих проектів.

Розпізнавання людського обличчя відіграє важливу роль у відеоспостереженні, інтерфейсі людина-комп'ютер, персоналізації різних додатків. У цій роботі ми представляємо підхід до виявлення та ідентифікації людського обличчя за відео у режимі реального часу, яке відстежує обличчя та порівнює його із збереженими даними. Наш підхід розпізнає людину протягом частки секунди, яка повністю ігнорує будь-який фоновий ефект. Крім того, цей метод також працює при різних умовах освітлення, що робить його придатним для виконання своєї мети в самих різних середовищах, не зазнаючи жодної суттєвої помилки. І цей підхід використовує переваги, пропоновані бібліотекою python, такою як OpenCV. Система управління базами даних SQLite використовується як магістраль її пам'яті. Завдяки своїй швидшій та автентичнішій роботі ця система є гідною для інтеграції з будь-якими додатками для підвищення автоматизації та зручності користування. OpenCV має досить потужний функціонал для детекту (визначення) об'єктів на зображенні.

Весь процес роботи системи найкраще пояснити трьома етапами. Етап збору даних – етап формування коду пітона в основному призначений для запису ідентифікаційних даних людини. В основному, це вказує камері зробити 20 зразків знімків. Коли дані людини збігаються з будь-яким попереднім записом, його/її зразки знімків оновлюються. Ця функція робить нашу систему пристосованою до невеликих змін у характері обличчя людини.

Тренувальний етап – цей етап в основному навчає систему, щоб мати можливість ідентифікувати конкретних осіб. Зразки зображень людини перетворюються у формат градацій сірого. Ця система не потребує кольору явно. Крім того, інформація про колір зображення іноді ускладнює ідентифікацію важливих країв. Тому колір на зображенні може розглядатися як шум у нашому дослідженні. Отже, використовується відтінок сірого. Отже, відтінок сірого робить алгоритм обробки зображень більш ефективним та швидшим. Коли зображення RGB перетворюється у формат градацій сірого, значення 3D-пікселів перетворюються на значення 1D. Як результат, нам доведеться мати справу з меншою кількістю даних, якщо використовується градація сірого.

Етап виявлення – камера подає на мікропроцесор відео в режимі реального часу. Відео імпортується до Python-коду мікропроцесора, як двовимірна матриця.

Спочатку частина, що містить обличчя, витягується із повного зображення, щоб зменшити ефект фону при обробці зображення, і зберігається в матриці. Потім матрицею маніпулюють і порівнюють із зразками фотографій для виявлення будь-якого збігу. Якщо знайдено будь-який збіг, на відеодисплеї в режимі реального часу відображається ім'я та додаткова інформація ідентифікованої особи із жовтим квадратом, що відстежує впізнане обличчя цієї людини. Додаткова інформація взята із системи управління базами даних SQLite.

Ефект нульового фону – у більшості випадків підхід до розпізнавання обличчя зазвичай сприятливий для місця, де беруться попередні дані. Як результат, він не працює автентично, коли тест запускається в новому середовищі. Це відбувається, коли загальна картина порівнюється із збереженими даними. Але жодного впливу фону на роботу цієї системи немає. Оскільки алгоритм витягує обличчя людини із зображень, що зберігаються в базі даних. Тому він працює ідеально, навіть незважаючи на зміну фону.

Програмно-апаратне забезпечення

Для здійснення налаштування, програмування потрібно насамперед мати відповідне програмне, а тако ж і апаратне забезпечення

- Raspberry Pi 4
- Блок живлення
- microSD
- клавіатура
- миша
- монітор
- кабель HDMI
- Камера

Рисунок 1 - Raspberry Pi 4b з радіаторним охолоджуючим корпусом та двома вентиляторами

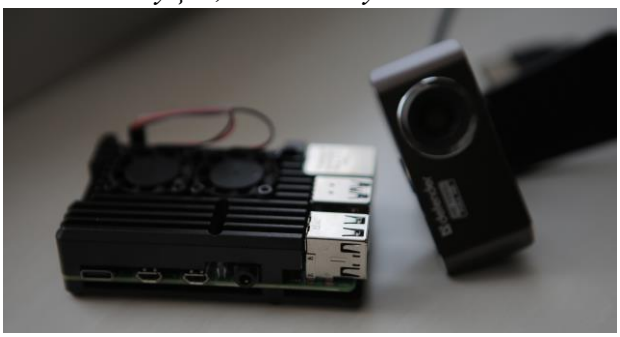

Рисунок 2 - Raspberry Pi 4b з USB web-камерою

Для розпізнавання обличчя Raspberry Pi ми будемо використовувати пакети OpenCV , face_recognition та imutils.

- **Raspbian** – основанная на Debian операционная система для Raspberry Pi.
- **OpenCV** – це бібліотека програмного забезпечення з відкритим кодом для обробки зображень і відео в режимі реального часу з можливостями машинного навчання.
- Ми будемо використовувати пакет **Python face_recognition** для обчислення обмежувального поля навколо кожного обличчя, обчислення вбудовування обличчя та порівняння граней у наборі даних кодування.
- **Imutils** – це низка зручних функцій для прискорення обчислень OpenCV на Raspberry Pi.

Встановлення залежностей для розпізнавання обличчя

Демонструємо алгоритм встановлення програмного забезпечення на одноплатний комп'ютер Raspberry PI. Для проведення симуляцій та навчання ми використовуємо Raspberry Pi 4b з 4 Gb RAM.

1. Підключаємо веб-камеру.
2. Завантажуємо Raspberry Pi.
3. Встановлюємо OpenCV.

```
sudo apt install cmake build-vital pkg-config git
sudo apt install libjpeg-dev libtiff-dev libjasper-dev libpng-dev libwebp-dev libopenexr-dev
sudo apt install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libxvidcore-dev libx264-dev
libdc1394-22-dev libgstreamer-plugins-base1.0-dev libgstreamer1.0-dev
sudo apt install libgtk-3-dev libqtgui4 libqtwebkit4 libqt4-test python3-pyqt5
sudo apt install libatlas-base-dev liblapacke-dev gfortran
sudo apt install libhdf5-dev libhdf5-103
sudo apt install python3-dev python3-pip python3-numpy
```

Продовжимо встановлення після розширення файлу підкачки.

Утилітою git clone (це утиліта командного рядка Git для вибору існуючого сховища з подальшим створенням його клону або копії) клонуємо opencv.git та opencv_contrib.git. Створюємо та переходимо в папку «build» в директорії «opencv» з подальшим виконанням коду:

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
….
make -j$(nproc)
```

4. Встановлюємо face_recognition.
5. Встановлюємо imutils (для розпізнавання об'єктів в реальному часі).

```
pip install imutils
```

Навчання моделі розпізнавання обличчя

На першому кроці ми формуємо DataSet – базу зображень чи фото, над якими буде проводитись робота. Для кожного користувача в базі створюється окрема директорія. Після підготовки зображень запускаємо скрипт, який проводить навчання системи та формує базу розпізнаних обличь.

```
cd facial_recognition
python train_model.py
```

Після опрацювання скрипта отримуємо карти координат лицьових точок для кожного фото (карти можуть бути формовані в залежності від кількості наявних фото). Більша кількість фото збільшує ймовірність розпізнання об'єкта з різних ракурсів.

```
python facial_req.py
```

При запуску веб-камери, починає працювати OpenCV, що визначає наявність обличчя в зображенні. Якщо результат позитивний, на виході отримуємо координати. Частина обличчя, що міститься в заданих межах порівнюється з сформованою базою і шукається збіг. В результаті на відео з'являється прямокутник та ім'я користувача.

Рисунок 3 – Скрін екрану під час розпізнавання користувача

Велике значення в таких системах відіграє швидкість розпізнавання, система розпізнавання обличчя в режимі реального часу повинна мати швидшу та ефективну швидкість обробки. Наша система досить швидка, і завершує весь процес розпізнавання мінімум три рази в секунду. У кращому випадку цикл виявлення завершується п'ять разів за секунду. Наше дослідження показує, що в середньому чотири цикли виявлення завершуються за одну секунду. Як результат, розпізнаванню зображень не перешкоджає, навіть якщо людина перебуває в русі. Ми використовували формат XML для збереження зображень у базі даних. Алгоритм призначений для вилучення лише обличчя людини. Отже, інша інформація про зображення, крім обличчя, не обробляється, а отже, час обробки зображень зменшується. Масштабування сірого також відіграє важливу роль для скорочення часу обробки. Як результат, виявити людину, яка рухається, набагато простіше.

Висновки.

Система побудована на Raspberry Pi, є мозком системи. У цій системі Python використовується для роботи з бібліотеками з відкритим кодом. Бібліотекою з відкритим кодом, яку ми використовували, є OpenCV, яка в основному спрямована на обробку комп'ютерного зору в режимі реального часу. Нерухоме зображення з відео зберігається у пітоні у вигляді двовимірного масиву. NumPy, є бібліотекою Python для обробки багатовимірного масиву, використовується для швидшої обробки нерухомого зображення. Система має систему управління сховищем, яка призначена для зберігання даних зацікавлених осіб. Для управління даними використовується популярна система управління базами даних SQLite. База даних містить інформацію про особу, яку слід виявити. Система є дуже корисною та ефективною, оскільки немає фонового ефекту для впізнання людини. Він може впізнати людину, навіть незважаючи на те, що середовище та фон змінюються, і задовільно працює в різних умовах освітлення. Невеликі зміни на обличчі людини, такі як макіяж, окуляри мало впливають на процес розпізнавання.

References.

1. S. Kostiuchko and V. Tchaban, "Variational Method of Auxiliary Equations in Nonlinear Systems Analysis and Synthesis Problems," *2019 IEEE 20th International Conference on Computational Problems of Electrical Engineering (CPEE)*, Lviv-Slavske, Ukraine, 2019, pp. 1-5, doi: 10.1109/CPEE47179.2019.8949123.
2. S. Kostiuchko, O. Kuzmych, A. Aitouche, S. Grinyuk and O. Mekush, "Application of Parametric Sensitivity Method to Analysis of Automatic Mooring Winch with Electric Drive System," *2019 4th Conference on Control and Fault Tolerant Systems (SysTol)*, Casablanca, Morocco, 2019, pp. 294-299, doi: 10.1109/SYSTOL.2019.8864751.

3. A. Pentland and T. Choudhury, "Face recognition for smart environments," Computer, vol. 33, no. 2, pp. 50–55, 2000. [2] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," Proceedings of the IEEE, vol. 83, no. 5, pp. 705–741, 1995.
4. C. Liu and H. Wechsler, "Evolutionary pursuit and its application to face recognition," IEEE Transactions on Pattern Analysis & Machine Intelligence, no. 6, pp. 570–582, 2000.
5. A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," IEEE Transactions on Pattern Analysis & Machine Intelligence, no. 12, pp. 1349–1380, 2000.
6. A. M. Burton, S. Wilson, M. Cowan, and V. Bruce, "Face recognition in poor-quality video: Evidence from security surveillance," Psychological Science, vol. 10, no. 3, pp. 243–248, 1999.
7. S. Eum, J. K. Suhr, and J. Kim, "Face recognizability evaluation for atm applications with exceptional occlusion handling," in Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on. IEEE, 2011, pp. 82–89.
8. T.-Y. Chen, C.-H. Chen, D.-J. Wang, and Y.-L. Kuo, "A people counting system based on face-detection," in Genetic and Evolutionary Computing (ICGEC), 2010 Fourth International Conference on. IEEE, 2010, pp. 699–702.
9. N. Kar, M. K. Debbarma, A. Saha, and D. R. Pal, "Study of implementing automated attendance system using face recognition technique," International Journal of computer and communication engineering, vol. 1, no. 2, p. 100, 2012.
10. S. Tiwari, A. Singh, and S. K. Singh, "Can face and soft-biometric traits assist in recognition of newborn?" in Recent Advances in Information Technology (RAIT), 2012 1st International Conference on. IEEE, 2012, pp. 74–79.