Санкт-Петербургский государственный университет
Направление подготовки «Прикладная математика - процессы управления»
Образовательная программа «Программирования и информационные технологии»

Отчёт по практическому заданию №2.
РЕАЛИЗАЦИЯ И ИССЛЕДОВАНИЕ ДЕТЕКТОРОВ ЛИЦ

по курсу “Прикладные задачи построения современных вычислительных систем”

Выполнил студент
группы 17.Б11-пу
Макоев Артур Аланович

Санкт-Петербург
2021

Содержание

1. Цель работы

Целью работы является создание и тестирование различных вариантов алгоритмов для распознавания лиц. В работе были рассмотрены:

- Алгоритм Template Matching
- Алгоритм Виолы-Джонса
- Алгоритм нахождения центральной и локальной линий симметрии лица

Исходный код программы находится по адресу github.com/zarond/BiometricTasks/Task2

Программа написана на языке Python, для графического интерфейса использовалась библиотека Tkinter, для математических вычислений библиотеки Numpy, SciPy, для работы с изображениями и видеопотоком использовались Pillow и OpenCV.

2. Задание №1

Для первого задания были созданы различные варианты алгоритма Template Matching, которые описаны в файле CustomFunctions.py, такие как:

- TemplateMatchingBasic – самая простая функция, вычисляющая выражение
$$R(x,y) = \sum_{x',y'} \left(I(x+x', y+y') - T(x',y')\right)^2$$
напрямую, с помощью двойного цикла for и np.sum(np.square(T-I)). Низкая производительность.
- CrossCorrelationBasic – простая функция, вычисляющая корреляцию сигналов по формуле
$$R(x,y) = \sum_{x',y'} I(x+x', y+y') \cdot T(x',y')$$
Тоже с помощью двойного цикла и np.sum(T*I), низкая производительность.
- CrossCorrelationFourier – функция корреляции, которая вычисляется путем свертки в пространстве коэффициентов Фурье. За счет преобразования сигналов в образы Фурье многократно повышается производительность свертки. В коде используется функция fftconvolution из математического пакета scipy
```
Res = signal.fftconvolve(X, np.flip(kernel), mode='valid')
```

Обратим внимание, что для использования формулы свертки при вычислении кросс корреляции необходимо отразить шаблон по обеим осям.

- TemplateMatchingFourier – вычисление суммы квадратов расстояний с применением операции свертки в прообразах Фурье. Делается это следующим образом:

```
corr = signal.fftconvolve(X, np.flip(kernel), mode='valid')
X1 = np.square(X)
G = np.sum(np.square(kernel))
kernel1 = np.ones(kernel.shape)
Res = signal.fftconvolve(X1, kernel1, mode='valid') + G - 2 * corr
```

Данный программный код вычисляет функцию:

$$R(x,y) = \sum_{x',y'} \left(I(x+x', y+y') - T(x',y')\right)^2$$

$$= \sum_{x',y'} \left(I^2(x+x', y+y') - 2 \cdot I(x+x', y+y') \cdot T(x',y') + T^2(x',y')\right) =$$

$$= \sum_{x',y'} I^2(x+x', y+y') - 2 \sum_{x',y'} I(x+x', y+y') \cdot T(x',y') + \sum_{x',y'} T^2(x',y') =$$

$$R(x,y) = \sum_{x',y'} I^2(x+x', y+y') - 2 \sum_{x',y'} I(x+x', y+y') \cdot T(x',y') + \sum_{x',y'} T^2(x',y')$$

Где в итоговом выражении $\sum_{x',y'} T^2(x',y') = const$ – не зависит от x,y и считается один раз, $\sum_{x',y'} I(x+x', y+y') \cdot T(x',y')$ - корреляция, которая считается через свертку Фурье, $\sum_{x',y'} I^2(x+x', y+y')$ - считается через свертку в коэффициентах Фурье между $I^2(x,y)$ – поэлементным возведением матрицы в квадрат и M – матрицей из единиц с размером, равным размеру шаблона T.

- CrossCorrelationMeanCorrectedFourier – корреляция, в которой в изображении и в шаблоне отняли их математическое ожидание.
- cv2.TM_SQDIFF, cv2.TM_SQDIFF_NORMED, cv2.TM_CCORR, cv2.TM_CCORR_NORMED, cv2.TM_CCOEFF, cv2.TM_CCOEFF_NORMED – эталонные методы из библиотеки OpenCV. TM_SQDIFF – формула template matching с суммой квадратов расстояний, cv2.TM_CCORR – корреляция, cv2.TM_CCOEFF – корреляция с центрированным математическим ожиданием, приставка NORMED означает, что метрика нормирована на диапазон [-1,1], то есть

$$R'(x,y) = \frac{R(x,y)}{\sqrt{\sum_{x',y'} T^2(x',y') \cdot \sum_{x',y'} I^2(x+x', y+y')}}$$

  Таким образом, cv2.TM_CCOEFF_NORMED – коэффициент корреляции Пирсона.

Стоит отметить, что алгоритмы в программе работают с однотонными изображениями, но для удобства в графическом интерфейсе сохраняются их цвета. Для распознавания с корреляцией необходимо найти точку максимума функции R(x,y), а для метода с суммой квадратов расстояний – точку минимума.

Рассмотрим саму программу:

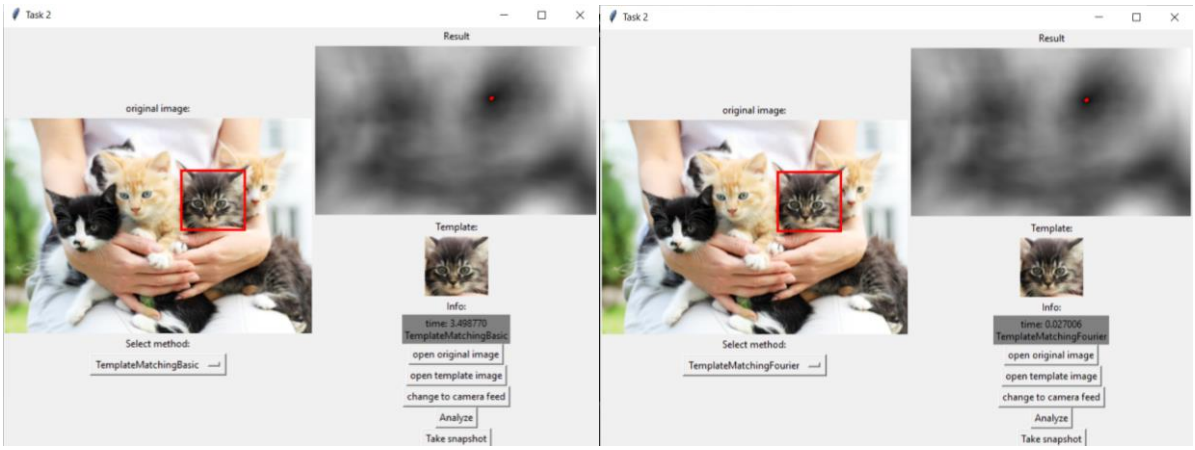


Рис. 1 Сравнение Template Matching с обычным вычислением и через свертку в образах Фурье. Второй вариант быстрее.

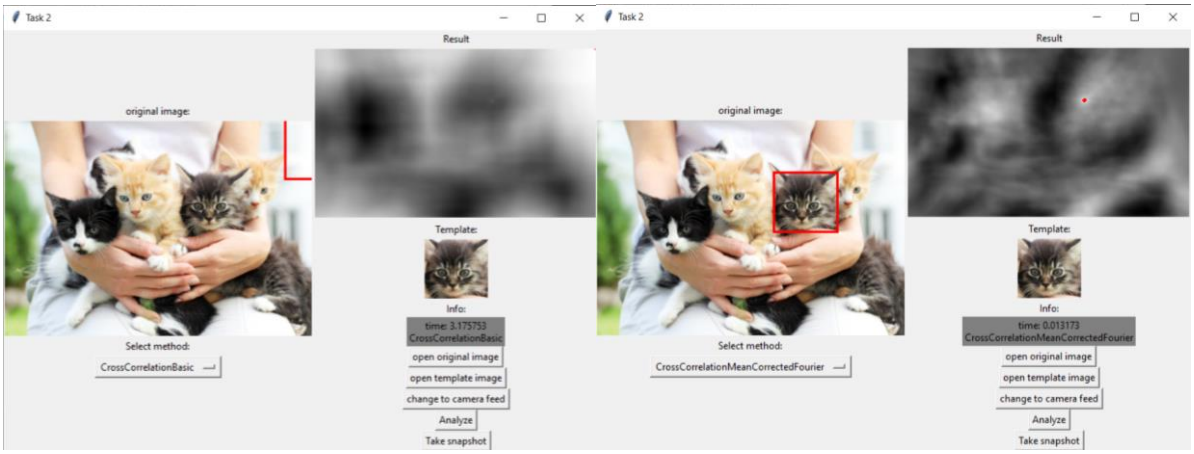


Рис. 2 Сравнение метода корреляции и корреляции с нулевым мат. ожиданием. Обычная корреляция дает ложное распознавание в ярких областях изображения.

Видно, что методы детекции с преобразованием Фурье во много раз быстрее прямых методов. Большинство методов справляется с нахождением части изображения в целом, кроме метода корреляции, который среагировал на самую яркую часть изображения. Улучшенная корреляция с центрированием по мат. ожиданию решила эту проблему.

В программе была реализована кнопка «auto mode», которая запускает режим распознавания лиц из базы данных ORLFaces и показывает результаты распознавания различными методами, меняя входное изображение каждые несколько секунд. Для выбора шаблона дается несколько вариантов в виде кнопок с изображениями. Можно использовать свой шаблон, если перед включением auto mode выбрать шаблон в основном окне.

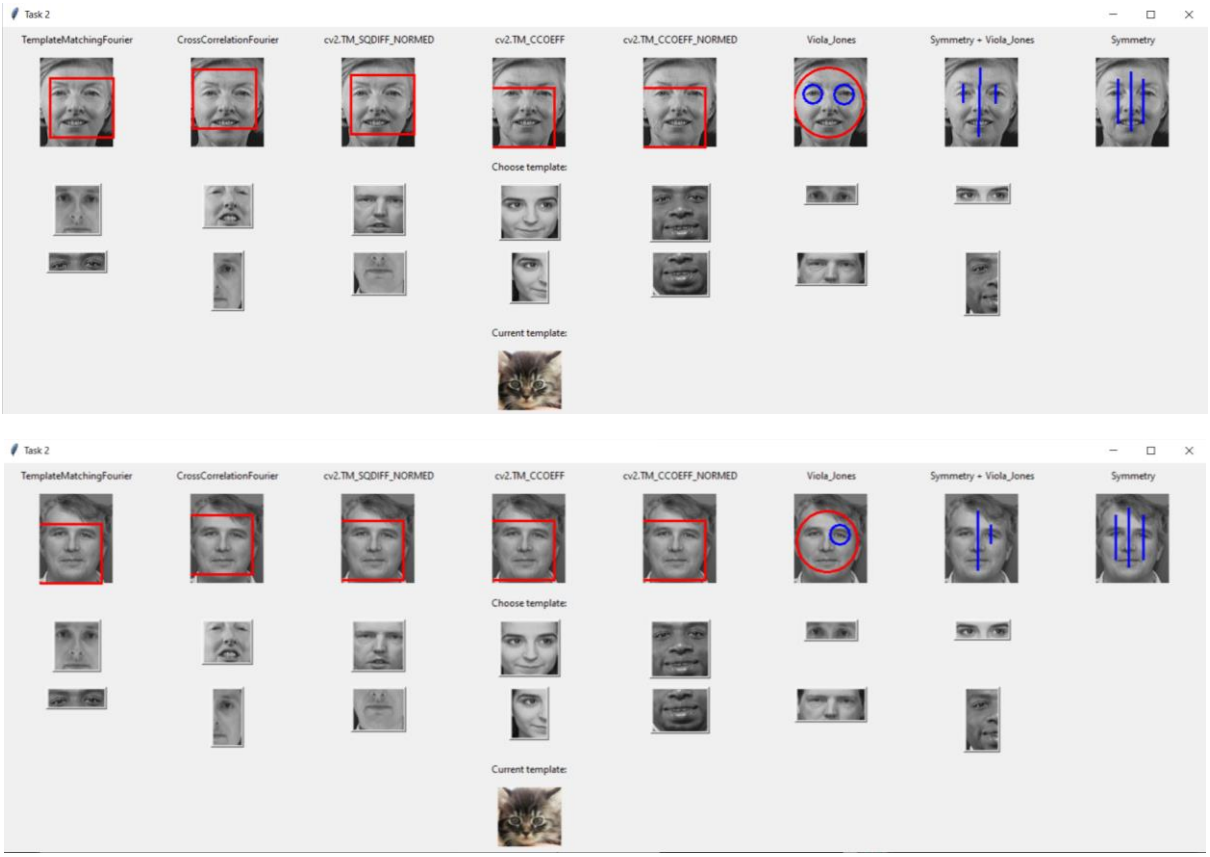


Рис. 3 Распознавание с шаблоном в виде кота.

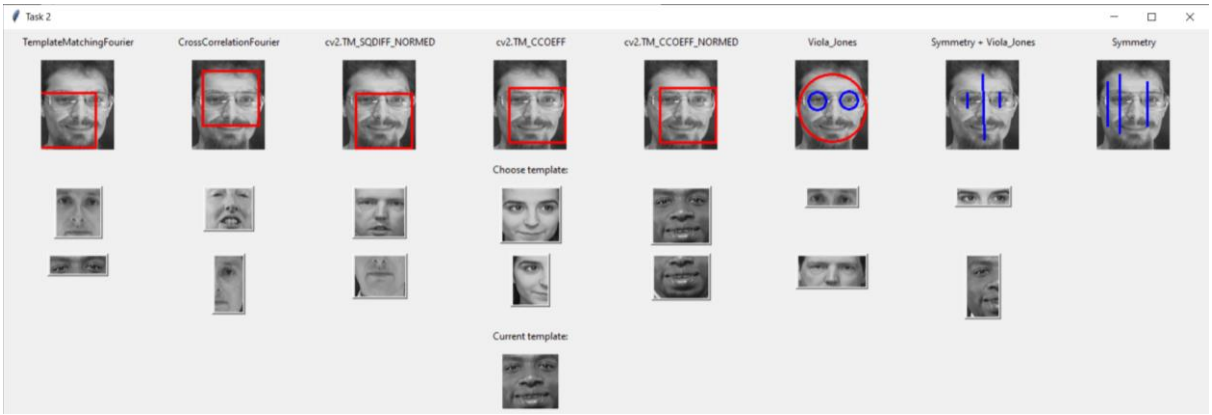
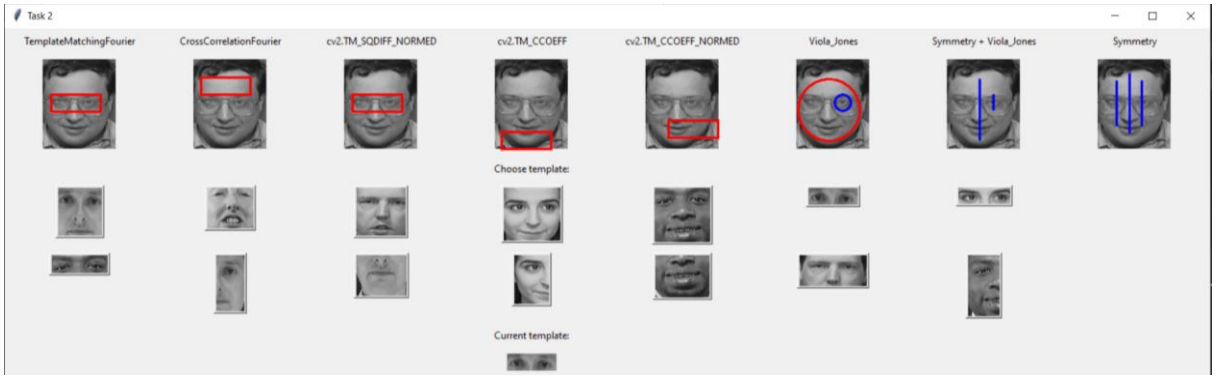


Рис. 4 Успешное распознавание с шаблоном - лицом африканца.



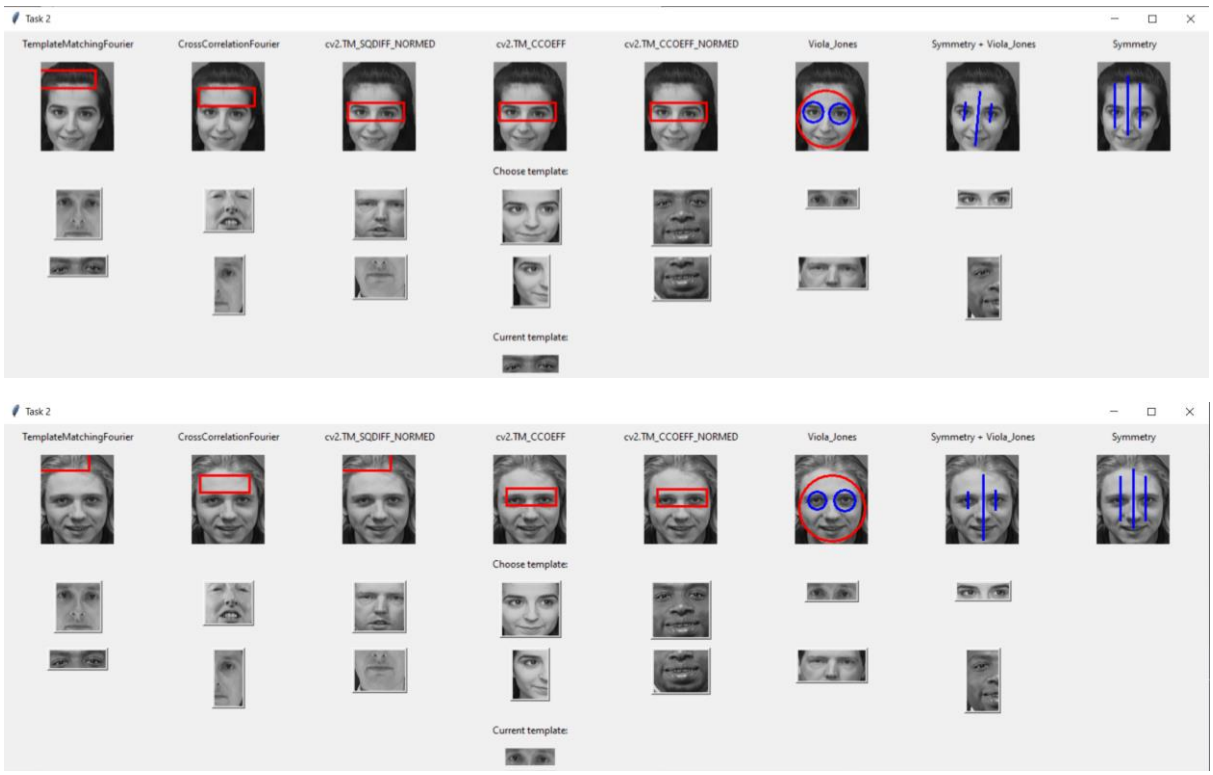


Рис. 5 Шаблон в виде глаз не очень удачно распознается корреляцией и суммой квадратов расстояний.

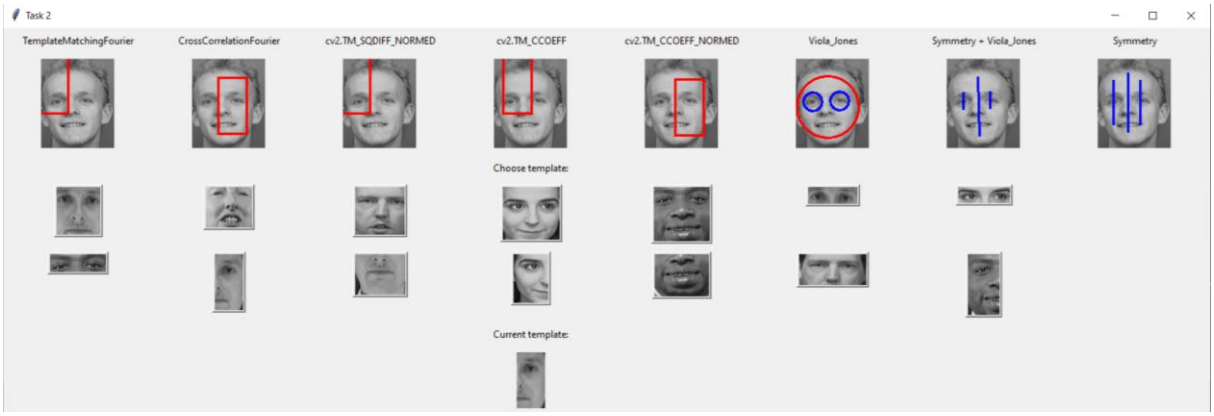
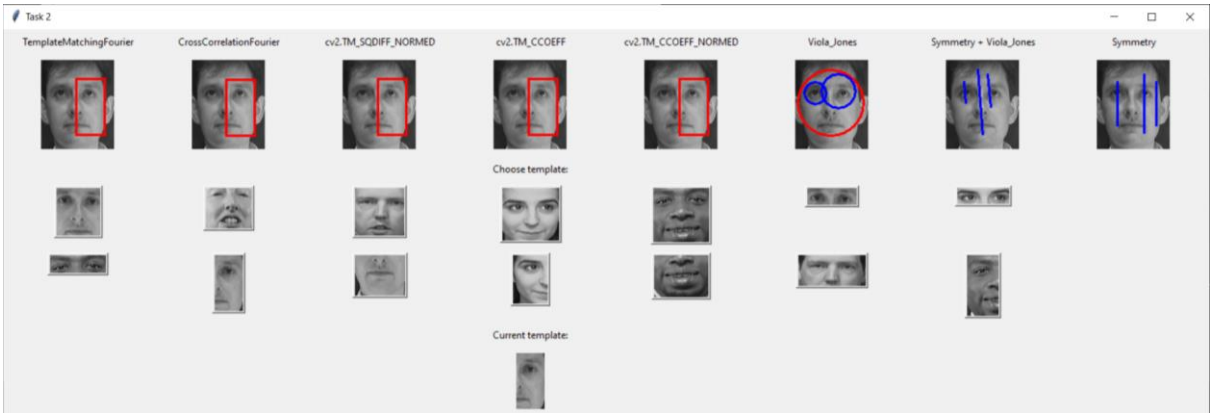


Рис. 6 Шаблон- правая часть лица.



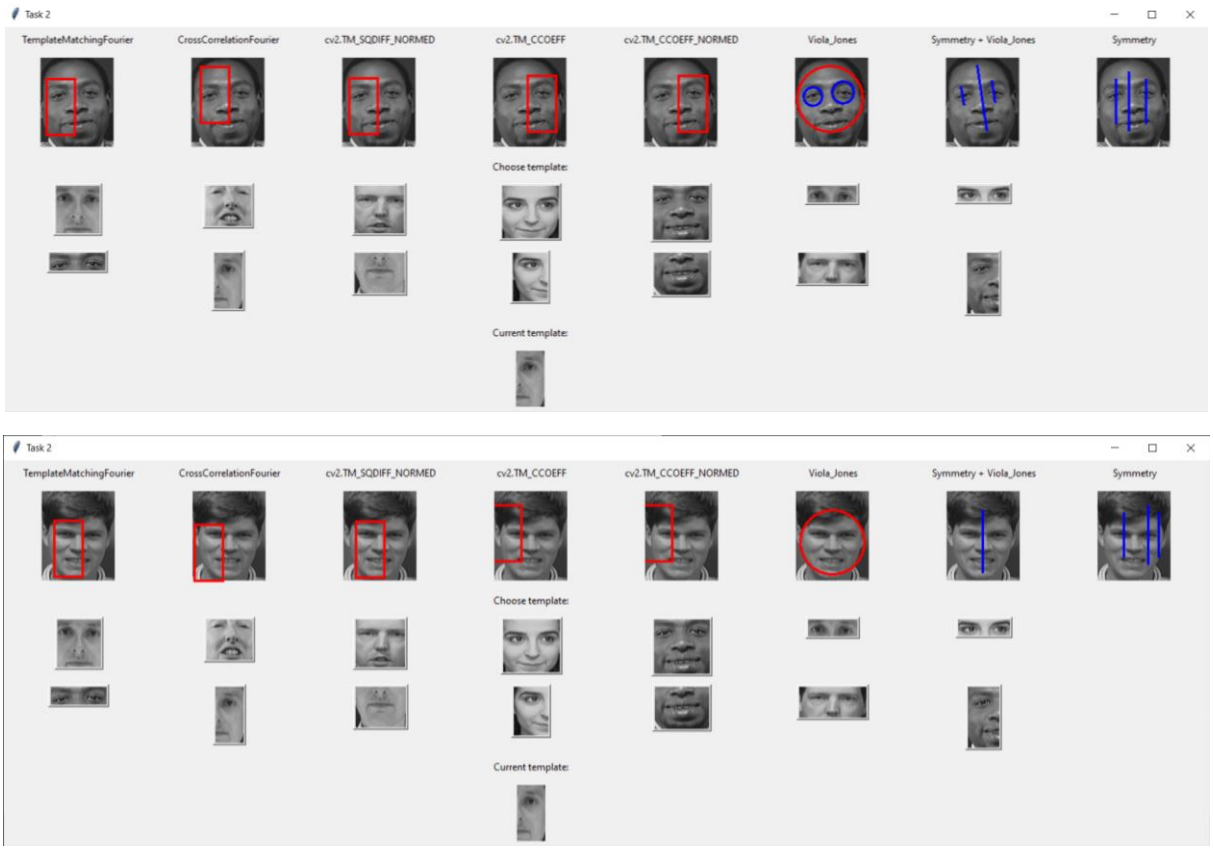
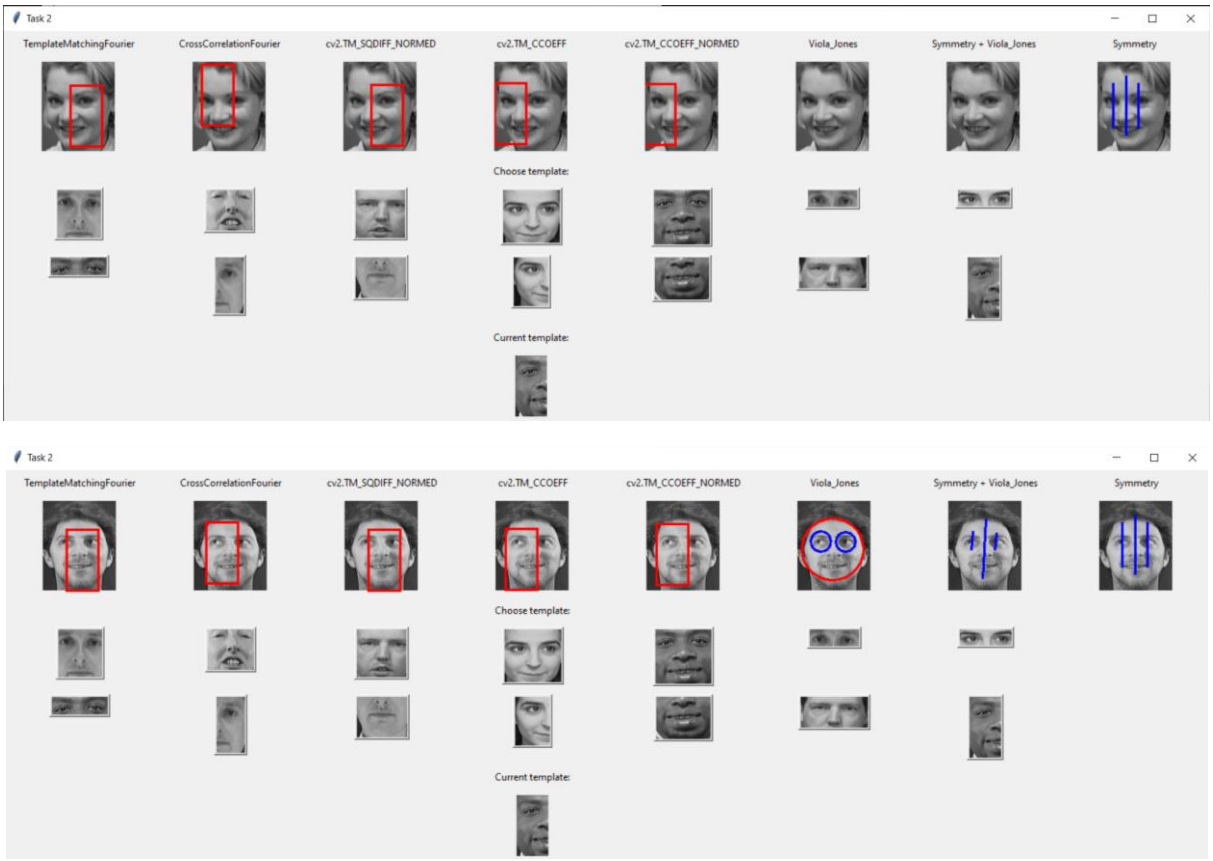


Рис. 7 Распознавание не всегда успешное.



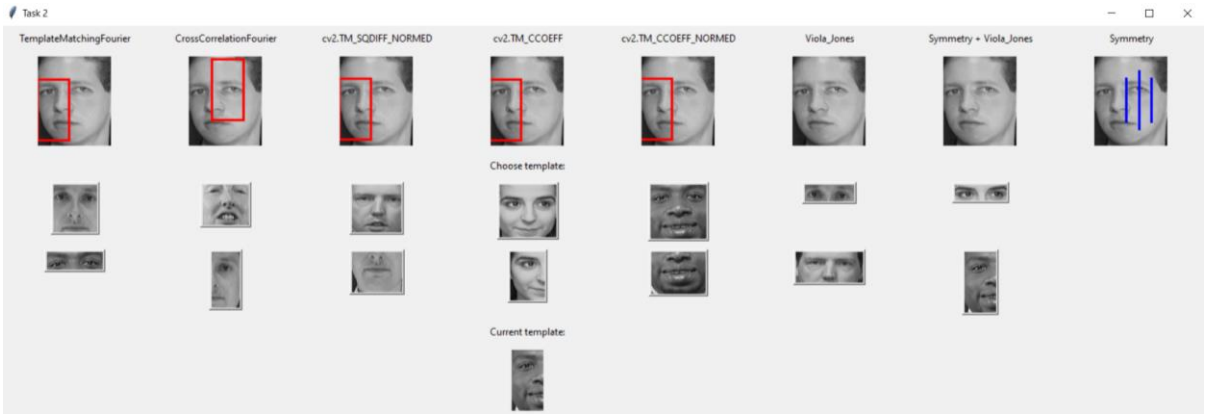


Рис. 8 Шаблон с африканцем может находить лица европейцев.

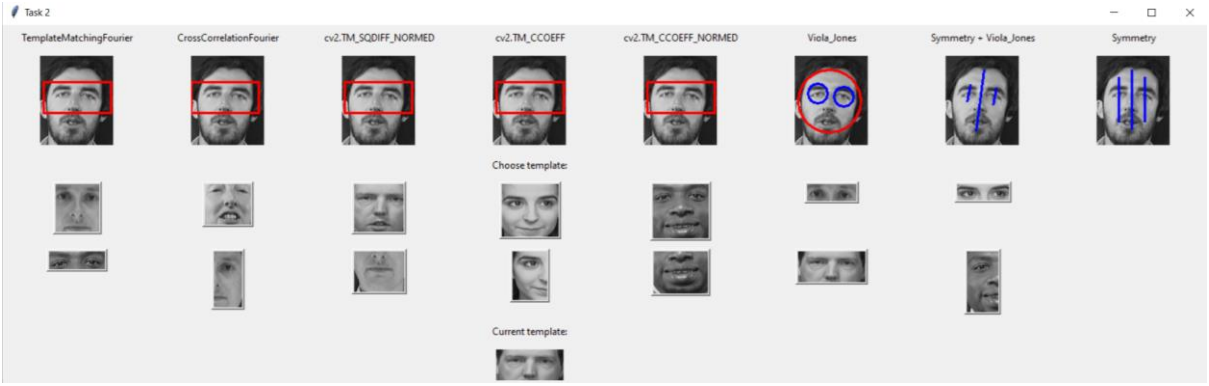


Рис. 9 Шаблон с глазами, бровями и носом - самый удачный.

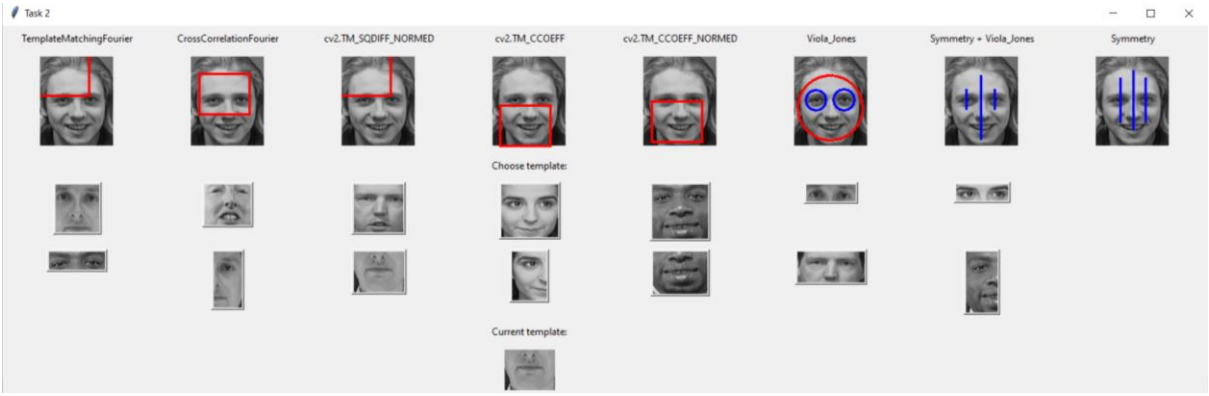


Рис. 10 Шаблон с одним ртом дает неточные результаты распознавания.

Вывод по алгоритмам Template matching: успешность распознавания очень сильно зависит от условий освещения и в исходном изображении, и в шаблоне. Лучше всего алгоритм работает, когда на шаблоне лицо того же человека, что и на фотографии. В случае, когда на шаблоне было лицо человека негроидной расы, детектор ни разу не смог поймать мое лицо. Детектор с шаблоном в полное лицо мог успешно распознавать лицо в некоторых пределах отдаления и поворотах. Шаблон с глазами чаще всего ложно детектировал лица на фоне, шаблон с нижней частью лица тоже обладал меньшей стабильностью, чем шаблон с полным лицом. Самым стабильным из неполных шаблонов оказался шаблон с правой половиной лица. С помощью этого шаблона детектировалась правая половина лица, и левая, в случае если правая скрыта. Этот шаблон оказался относительно устойчив к повороту головы.

Другой интересный эффект, который я заметил, связан с тем, что иногда изображение с веб камеры сильно меняло экспозицию, делая лицо засвеченным, а комнату более яркой. Это мешало работе моего алгоритма, но интересен тот факт, что часто сначала мой алгоритм соскакивал с лица на фон, а затем уже камера меняла экспозицию. При условии, что в своей программе я не управляю экспозицией камеры, это значит, что в драйвер веб камеры встроено распознавание лиц для того, чтобы управлять экспозицией и фокусироваться на лицах. И этот алгоритм ломался примерно при тех же условиях, что и мой – при сильных поворотах и вращении головы и закрытии большей части лица.

По моему мнению, чтобы успешно распознавать лицо необходима оценка сразу по нескольким шаблонам: например, по полному лицу и по двум половинам лица. В этом случае увеличивается шанс верного распознавания при повороте головы, когда видна только половина лица. В качестве дополнительных шаблонов можно рассматривать глаза и рот, или пользоваться отдельными шаблонами для различных углов вращения головы.

Для задачи распознавания лица в видеопотоке неплохой мыслью будет выбор из нескольких локальных экстремумов того, который ближе к положению лица из предыдущего кадра, чтобы фильтровать шум. В качестве шаблона можно брать успешно распознанное ранее лицо.

3. Задание №2

В данном задании я исследовал детектор Виолы-Джонса. Использовал реализацию из библиотеки OpenCV.

```
self.face_cascade = cv2.CascadeClassifier()

frame_gray = cv2.equalizeHist(np.array(X,dtype='uint8'))

face_cascade.detectMultiScale(frame_gray)
```

Для работы детектора необходимы файлы с настройками каскадов для распознавания. Использовались следующие файлы:

```
face_cascade_name = 'data/haarcascade_frontalface_default.xml'
eyes_cascade_name = 'data/haarcascade_eye.xml'
```

Я дополнительно сортировал найденные лица по размеру и использовал только самое большое найденное лицо. Я ограничил распознавание только двумя самыми большими глазами. Однако алгоритм все равно давал шумовые результаты. Часто он совсем терял лицо и выдавал случайные распознавания на фоне. У алгоритма была проблема с распознаванием обоих глаз одновременно, глаза распознавались на месте ноздрей и рта. Детектор плохо работал при повороте головы.

4. Задание №3

В моей программе есть два типа алгоритма для определения симметрии.

Первый алгоритм работает вместе с детектором Виолы-Джонса, что позволяет заранее знать положение и размеры головы и глаз. В этом случае определение оси симметрии работает на основе template matching. В качестве шаблона берется отзеркаленное найденное лицо, а в качестве изображения берется то же лицо, но в оригинальном положении и с дополнением в виде нулей на границах изображения шириной с само изображение. В случае, когда детектор определяет успешно оба глаза, то становится известен угол поворота головы и можно повернуть изображение, прежде чем применить поиск оси симметрии, получив таким образом более точное положение оси. Тот же алгоритм применяется и для нахождения осей симметрии глаз.

Второй алгоритм работает без информации о положении лица и глаз. Сначала находится центральная ось симметрии, а затем в разделенном на две половины лице находятся локальные линии симметрии. В этом случае применяется алгоритм, считающий градиент в горизонтальном направлении, сравнивая расстояние между двумя зеркально расположенными окнами по две стороны от сканирующей прямой, которая движется по изображению слева направо.

Алгоритмы нахождения оси симметрии может хорошо срабатывать для головы, повернутой набок, но он начинает быстро терять точность при повороте головы в профиль. Второй алгоритм иногда может принять локальную ось симметрии за центральную. Нахождению локальной оси симметрии могут помешать очки.

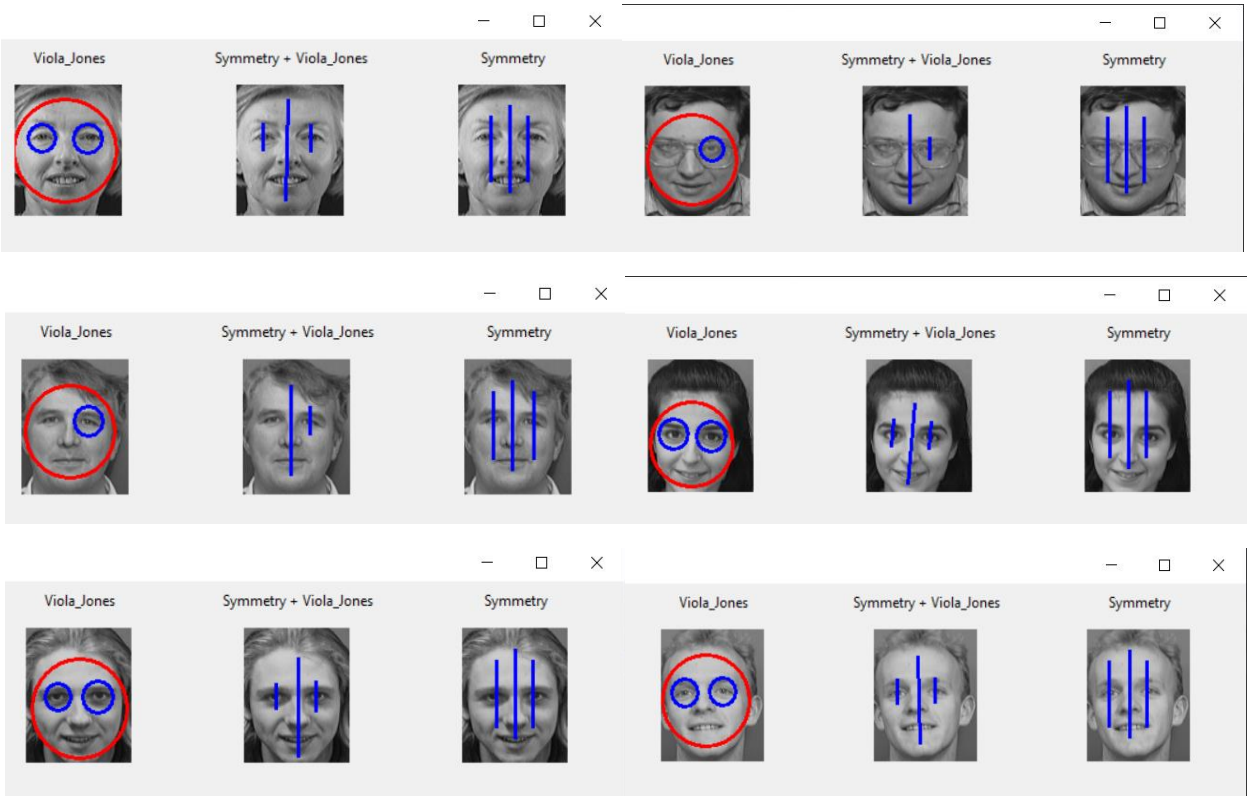


Рис. 11 Удачные распознавания линий симметрии. Левая картинка - результат работы алгоритма Виолы-Джонса, средняя - первый алгоритм нахождения симметрии, правая – второй алгоритм.

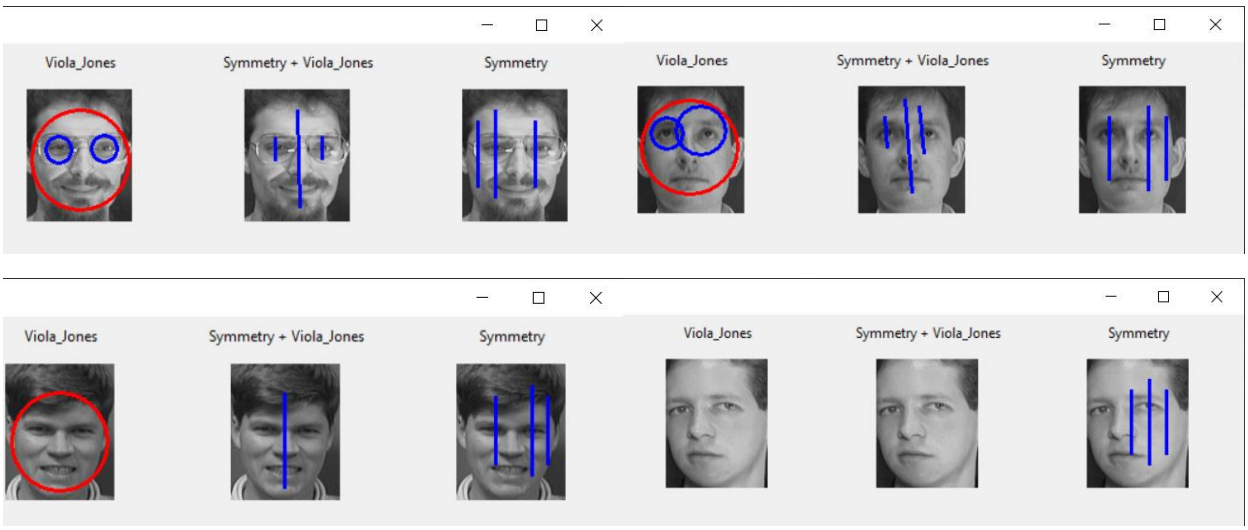


Рис. 12 Неудачные результаты нахождения симметрии.

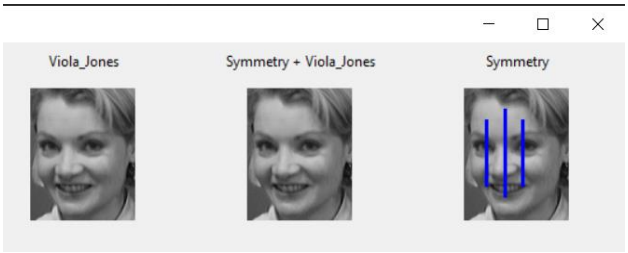


Рис. 13 Второй алгоритм справился в ситуации, когда метод Виолы-Джонса не распознал лицо.

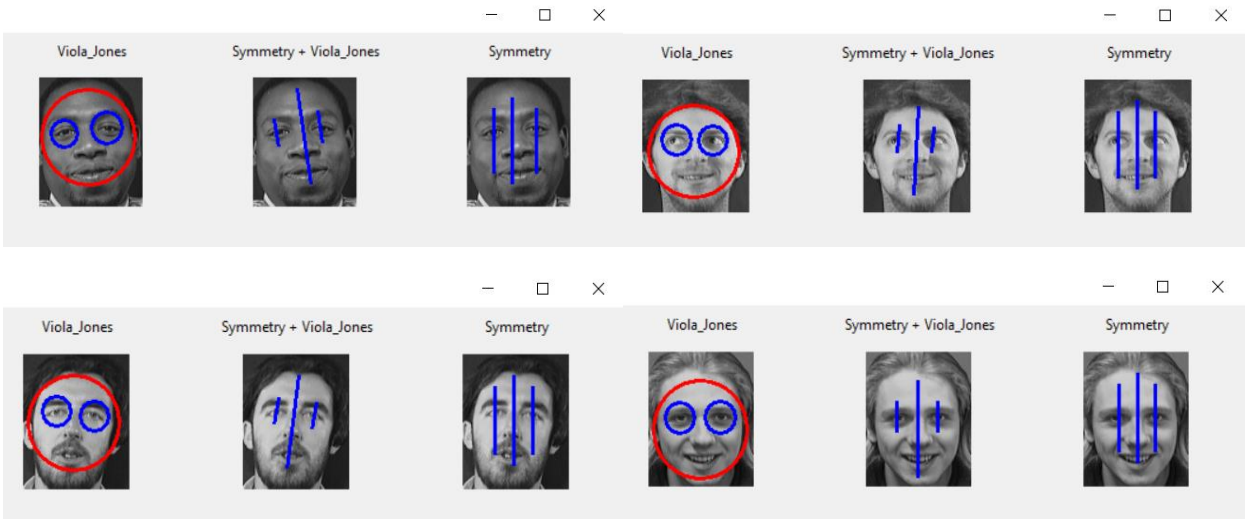


Рис. 14 Примеры удачной работы алгоритма.

5. Заключение

В данной работе я рассмотрел несколько алгоритмов для детектирования лиц на фотографиях. Среди трех опробованных вариантов: template matching, SHORE от Fraunhofer и детектора Виола-Джонса, лучший результат показало программное обеспечение от Fraunhofer. К сожалению, я не нашел в интернете на чем основан их метод, только его

название - Sophisticated High-speed Object Recognition Engine. Статья об алгоритме на сайте researchgate.net не открыта для общего доступа.

Точность алгоритма распознавания на основе template matching сильно зависит от освещения и соответствия шаблона изображению, часто выдает ложноположительные результаты. Алгоритм Виолы-Джонса меньше зависит от условий освещения и лучше срабатывает при разных размерах лица на изображении, но распознавание подвержено шумам. Алгоритм нахождения симметрии сбоит, если были неправильно определены положения глаз. Он хорошо работает в анфас, но при повороте в профиль симметрии как таковой уже не наблюдается.

Каждый алгоритм обладает своими недостатками, и для создания продвинутой системы детектирования лиц нельзя полагаться на какой-то один шаблон, желательно использовать сильные стороны алгоритмов там, где это возможно, и не использовать алгоритм в условиях, где он может показать себя неэффективным.