

Raport z testów bibliotek OpenCV oraz OpenIMAJ

SPIS TREŚCI

Spis treści	2
1. Wstęp.....	3
2. Klasyfikatory Haara	3
3. Sposoby wykrywania uśmiechu.....	3
4. Zdjęcia zastosowane w trakcie testów	4
5. Sposób przeprowadzenia testów.....	5
6. Wyniki testów.....	5
6.1. OpenCV	5
6.1.1. Wykrywanie twarzy.....	5
6.1.2. Wykrywanie uśmiechu z wykorzystaniem wyłącznie klasyfikatora uśmiechu	6
6.1.3. Wykrywanie uśmiechu przy pomocy sposobu wykrywania dolnej części	6
twarzy	
6.2. OpenIMAJ.....	7
6.2.1. Wykrywanie twarzy.....	7
6.2.2. Wykrywanie uśmiechu z wykorzystaniem wyłącznie klasyfikatora uśmiechu	8
6.2.3. Wykrywanie uśmiechu przy pomocy sposobu wykrywania dolnej części	8
twarzy	
7. Omówienie wyników	9
8. Podsumowanie	10
9. Literatura.....	11

1. WSTĘP

W trakcie wstępnego rozpoznania do realizacji problemu rozpoznawania uśmiechu udało się znaleźć trzy główne biblioteki realizujące detekcję fragmentów obrazu w języku *Java*: *OpenCV*, *JavaCV* oraz *OpenIMAJ*. Według uzyskanych informacji *JavaCV* jest nakładką na *OpenCV*, dlatego też zdecydowano się odstąpić od testów tej biblioteki – w jej nowszych wersjach istnieje wbudowana możliwość uruchamiania biblioteki z poziomu *Javy*. Testom sprawdzającym możliwości oraz ogólną wydajność poddano zatem *OpenCV* oraz *OpenIMAJ*. Obie wykorzystują tzw. klasyfikatory Haara do wskazania interesujących fragmentów obrazu.

Testy zostały przeprowadzone w prostej aplikacji *Java*, która przyjmuje na wejście listę zdjęć, które mają zostać poddane algorytmom wyszukiwania uśmiechu.

2. KLASYFIKATORY HAARA

Klasyfikator Haara jest plikiem *XML*, który informuje mechanizm wykrywania fragmentów obrazu, czego powinien na nim szukać. Najpopularniejszymi klasyfikatorami są: wykrywanie twarzy z przodu oraz z profilu, detekcja oczu oraz ust. Niestety, chociaż obie testowane biblioteki są dosyć popularne i łatwo można znaleźć sporą liczbę klasyfikatorów wygenerowanych przez społeczność [1] lub nawet twórców biblioteki [2], nie udało się odnaleźć niezawodnego klasyfikatora do wykrywania uśmiechu – są one w dużym stopniu mylne i bardzo często wykrywają oczy lub po prostu usta bez widocznego uśmiechu.

W trakcie poszukiwań udało się odnaleźć informacje na temat sposobu wytrenowania własnego klasyfikatora – proces w skrócie ogranicza się do znalezienia próbek pozytywnych, zdjęć z widocznymi uśmiechami oraz próbek negatywnych, w których nie znajduje się żaden uśmiech. Ponieważ jednak jest on dosyć czasochłonny oraz wymaga zastosowania sporej liczby zdjęć, do testów biblioteki zdecydowano się wykorzystać dwa klasyfikatory: twarzy z przodu oraz uśmiechu/ust, znalezione na wspomnianych wcześniej zasobach internetowych.

3. SPOSOBY WYKRYWANIA UŚMIECHU

Istnieje kilka sposobów wykrywania uśmiechu na obrazie – różnią się one poziomem skomplikowania algorytmu. Najprostszym z nich jest zastosowanie klasyfikatora uśmiechu na całym obrazie – wymaga on jednak wysokiej skuteczności działania, a więc też odpowiedniego przygotowania próbek do zadanego problemu. Sposób ten ma duże ryzyko wystąpienia tzw. *false alarmów*, czyli wykrywania uśmiechu w miejscach, które nie powinny być możliwe do wystąpienia. Typowymi przykładami takiej sytuacji są: uśmiech na ścianie, ramieniu czy szafie.

Ulepszeniem takiego podejścia jest zauważenie faktu, że uśmiech powinien występować wyłącznie na ludzkiej twarzy. W tym sposobie należy wstępnie odnaleźć twarze na obrazie, wyciąć te fragmenty zdjęcia, a następnie wewnątrz nich wyszukać uśmiechu. Dzięki temu eliminuje się sporo problemów z nieprawidłowym wykryciem uśmiechu – eliminuje się

ryzyko wystąpienia uśmiechu np. na ścianie. Wciąż jednak występuje ryzyko wskazania oczu jako uśmiechniętych ust.




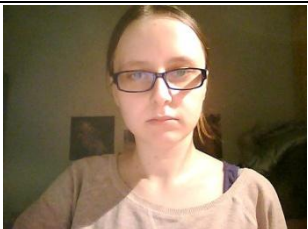


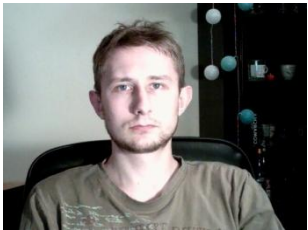

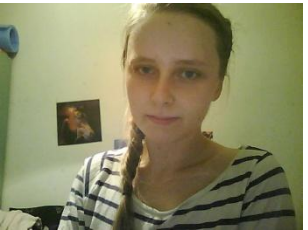
Inny sposób opisuje Amine Sehili na swoim blogu [3], na którym wymienia problemy z poprzednim podejściem: wykrywane były jego okulary, czasem nawet zawodziła detekcja twarzy. Według niego, należy na początku odszukać twarze, a następnie wewnątrz nich odnaleźć nos i odrzucić fragmenty, które go nie zawierają. Kluczowym momentem tego algorytmu jest zauważenie faktu, że usta występują tuż pod nosem – można zatem odrzucić wyniki, które znajdują się nad nim.




W trakcie testów zdecydowano się porównać zwykłe zastosowanie klasyfikatora wyszukującego uśmiech z uproszczonym ostatnim sposobem, czyli wyszukiwaniem ust jedynie w dolnej połowie twarzy.

4. ZDJĘCIA ZASTOSOWANE W TRAKCIE TESTÓW

W ramach testów przygotowano 12 zdjęć z kamery internetowej 4World Webcam Z200 (2Mpix) oraz wbudowanej, o rozdzielczości 640x480, ponieważ taka forma obrazów będzie docelowo wykorzystywana w projekcie. Zdjęcia przeprowadzono w różnych warunkach (słabe i dobre oświetlenie, okulary i bez okularów) oraz na osobnikach obu płci. Dodatkowo, zdjęcia zawierają różne poziomy uśmiechu. *Tabela 1. Zdjęcia wykorzystane w trakcie testów* pokazuje spis zdjęć wykorzystanych w trakcie testów.

Tabela 1. Zdjęcia wykorzystane w trakcie testów

		
Z1. Średnie oświetlenie, brak uśmiechu, mężczyzna	Z2. Średnie oświetlenie, średni uśmiech, mężczyzna	Z3. Średnie oświetlenie, widoczny uśmiech, mężczyzna
		
Z4. Kobieta, brak uśmiechu	Z5. Kobieta, średni uśmiech	Z6. Kobieta, widoczny uśmiech
		

Z7. Dobre oświetlenie, mężczyzna, brak uśmiechu	Z8. Dobre oświetlenie, mężczyzna, uśmiech	Z9. Kobieta, brak uśmiechu
		
Z10. Mężczyzna w okularach, dobre oświetlenie, brak uśmiechu	Z11. Mężczyzna w okularach, dobre oświetlenie, widoczny uśmiech	Z12. Uśmiechnięta kobieta

5. SPOSÓB PRZEPROWADZENIA TESTÓW

Testy przeprowadzono przy użyciu programu napisanego w języku *Java*. Aplikacja na wejście otrzymuje listę zdjęć, które mają zostać zweryfikowane pod kątem wystąpienia twarzy i uśmiechu. Dla każdego zdjęcia aplikacja robi pomiar czasu wykrywania twarzy, uśmiechu przy użyciu tylko klasyfikatora uśmiechu oraz uśmiechu przy użyciu sposobu wykrywania dolnej części twarzy. Każde wykrycie zapisywane jest na dysku w formie obrazka – fioletowa ramka oznacza wykrytą twarz, zielona natomiast – wykryty uśmiech. Testy przeprowadzono przy użyciu bibliotek *OpenCV* oraz *OpenIMAJ*. Parametryzację detekcji w obu bibliotekach pozostawiono domyślną.

W czas detekcji nie jest wliczany czas naniesienia prostokątów na obraz i zapisanie go na dysku.

6. WYNIKI TESTÓW

Wyniki wykrywania twarzy przedstawiono w tabelce jedynie w postaci czasów wykrywania, ponieważ są one pokazane w pozostałych dwóch testach, jako fioletowe prostokąty.

W teście wykrywania uśmiechu przedstawiono wyniki (zielone prostokąty) wykrywania uśmiechu oraz czas, jaki zajął ten proces.

6.1. *OpenCV*

6.1.1. *Wykrywanie twarzy*

Tabela 2. Detekcja twarzy w *OpenCV*

Z1	Z2	Z3	Z4
86 ms	87 ms	83 ms	95 ms
Z5	Z6	Z7	Z8
106 ms	96 ms	92 ms	90 ms

Z9	Z10	Z11	Z12
90 ms	91 ms	96 ms	88 ms

6.1.2. Wykrywanie uśmiechu z wykorzystaniem wyłącznie klasyfikatora uśmiechu



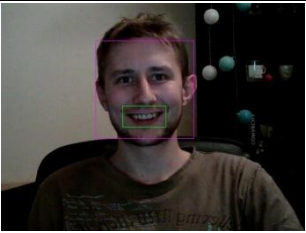
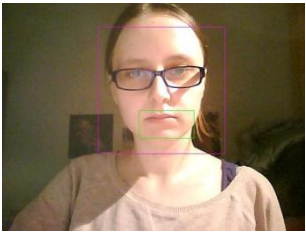
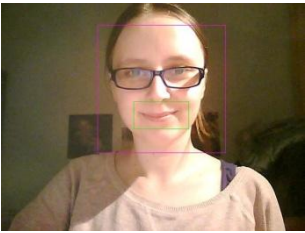
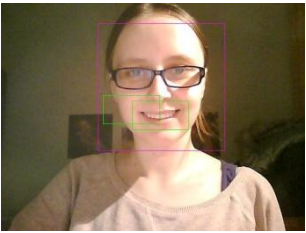
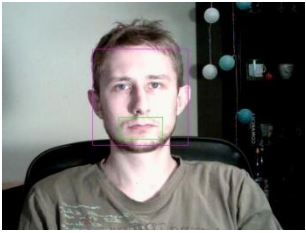
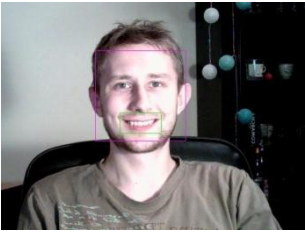
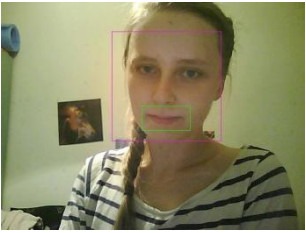
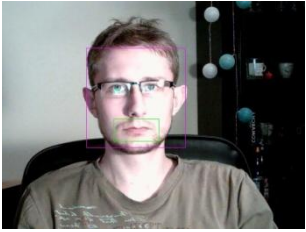
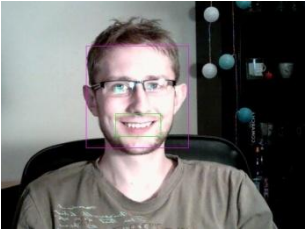
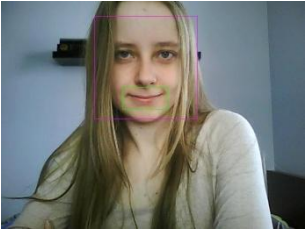
Tabela 3. Wykrywanie uśmiechu z wykorzystaniem klasyfikatora w OpenCV

Z1	Z2	Z3
		
228 ms	250 ms	248 ms
Z4	Z5	Z6
		
248 ms	471 ms	286 ms
Z7	Z8	Z9
		
199 ms	225 ms	216 ms
Z10	Z11	Z12
		
219 ms	197 ms	251 ms

6.1.3. Wykrywanie uśmiechu przy pomocy sposobu wykrywania dolnej części twarzy

Tabela 4. Wykrywanie uśmiechu z użyciem wykrywania twarzy w OpenCV

Z1	Z2	Z3
-----------	-----------	-----------

		
36 ms	40 ms	42 ms
Z4	Z5	Z6
		
38 ms	40 ms	41 ms
Z7	Z8	Z9
		
40 ms	45 ms	44 ms
Z10	Z11	Z12
		
44 ms	42 ms	41 ms

6.2. OpenIMAJ

6.2.1. Wykrywanie twarzy

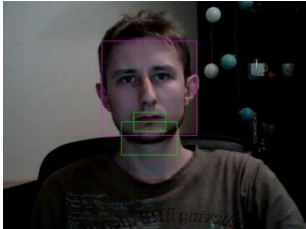
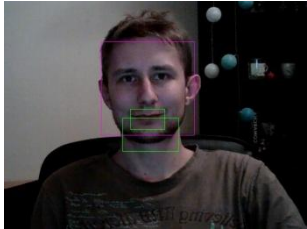
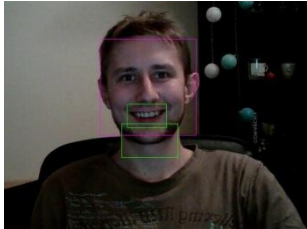
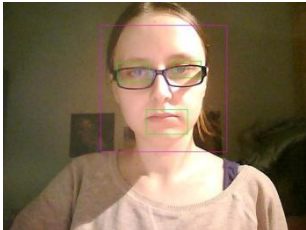
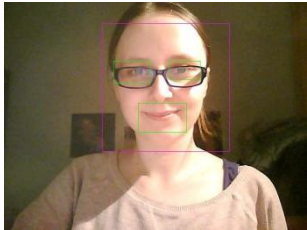
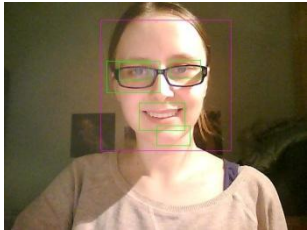
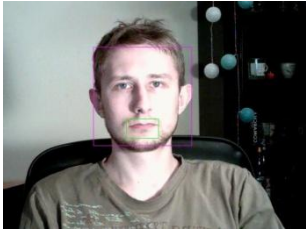

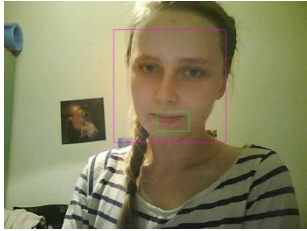
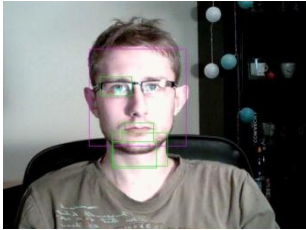
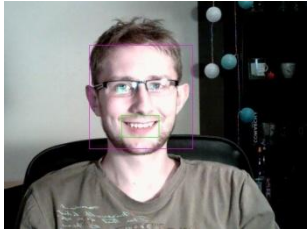
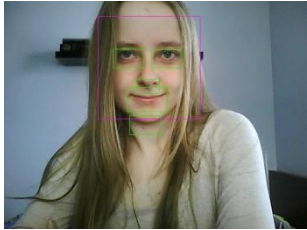
Tabela 5. Detekcja twarzy w OpenIMAJ

Z1	Z2	Z3	Z4
445 ms	451 ms	446 ms	581 ms
Z5	Z6	Z7	Z8
585 ms	573 ms	484 ms	482 ms
Z9	Z10	Z11	Z12

448 ms	482 ms	500 ms	502 ms
--------	--------	--------	--------

6.2.2. Wykrywanie uśmiechu z wykorzystaniem wyłącznie klasyfikatora uśmiechu



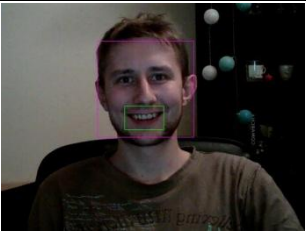
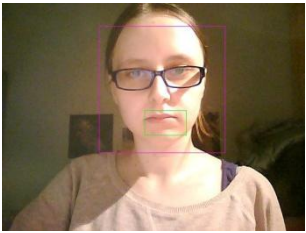
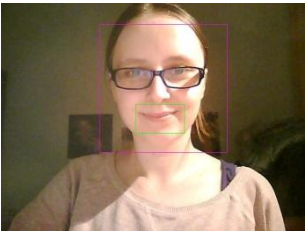
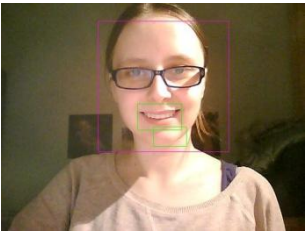
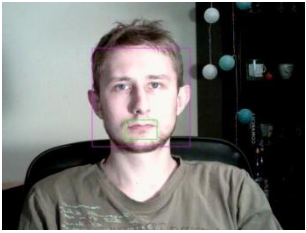

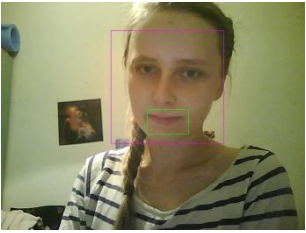

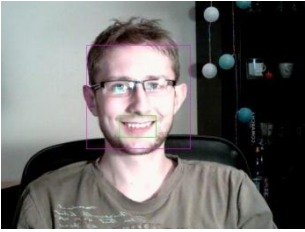

Tabela 6. Wykrywanie uśmiechu z wykorzystaniem klasyfikatora w OpenIMAJ

Z1	Z2	Z3
		
215 ms	216 ms	218 ms
Z4	Z5	Z6
		
258 ms	286 ms	260 ms
Z7	Z8	Z9
		
230 ms	232 ms	239 ms
Z10	Z11	Z12
		
237 ms	239 ms	231 ms

6.2.3. Wykrywanie uśmiechu przy pomocy sposobu wykrywania dolnej części twarzy

Tabela 7. Wykrywanie uśmiechu z użyciem wykrywania twarzy w OpenIMAJ

Z1	Z2	Z3
----	----	----

		
483 ms	477 ms	482 ms
Z4	Z5	Z6
		
659 ms	643 ms	652 ms
Z7	Z8	Z9
		
525 ms	523 ms	528 ms
Z10	Z11	Z12
		
549 ms	610 ms	580 ms

7. OMÓWIENIE WYNIKÓW

Zaprezentowane wyniki pokazują, że w obu przypadkach klasyfikatory uśmiechu okazały się być zbyt proste, by rozróżnić uśmiech od ust.

Jak można zauważyć z tabeli *Tabela 2. Detekcja twarzy w OpenCV*, czas średni, jaki potrzebuje biblioteka *OpenCV* na proste wykrycie twarzy wynosi *91 ms*. Do tego samego zadania *OpenIMAJ* (*Tabela 5. Detekcja twarzy w OpenIMAJ*) potrzebuje średnio aż *498 ms*, przy czym wyniki na zdjęciach wydają się być porównywalne – obie biblioteki nie popełniały błędów przy znajdowaniu twarzy.

Nieco inaczej prezentuje się sytuacja w przypadku detekcji uśmiechu przy pomocy prostego klasyfikatora (*Tabela 3. Wykrywanie uśmiechu z wykorzystaniem klasyfikatora w OpenCV* oraz *Tabela 6. Wykrywanie uśmiechu z wykorzystaniem klasyfikatora w OpenIMAJ*). Na pierwszy rzut oka można zauważyć, że klasyfikator w *OpenCV* przy użyciu domyślnych parametrów prezentuje kompletnie bezużyteczne wyniki w czasie średnim 253 ms. *OpenIMAJ* okazało się w tym przypadku znacznie lepsze – myli jedynie się w przypadku podbródków i oczu, a do zaprezentowania wyników potrzebuje średnio 238 ms.

W przypadku wykrywania uśmiechu z użyciem algorytmu wykrywania twarzy (*Tabela 4. Wykrywanie uśmiechu z użyciem wykrywania twarzy w OpenCV* oraz *Tabela 7. Wykrywanie uśmiechu z użyciem wykrywania twarzy w OpenIMAJ*) znacznie lepsze okazało się być *OpenCV*. Usta zostały wykryte poprawnie w większości przypadków, w czasie średnim 41 ms. Biblioteka *OpenIMAJ* również z bardzo wysoką dokładnością wskazała w tym przypadku usta, jednak czas, w jakim zwróciła wyniki, jest dużo większy – aż 559 ms.

Z przedstawionych wyników można wywnioskować, że biblioteka *OpenCV* jest znacznie wydajniejsza od *OpenIMAJ* – dzieje się tak przez bezpośrednie wykonywanie obliczeń detekcji w języku C++, podczas gdy *OpenIMAJ* jest w całości napisane w *Javie*. Można jednak łatwo zauważyć, że przy zastosowaniu prostych metod, *OpenIMAJ* sprawdza się lepiej – dzieje się tak prawdopodobnie przez niską jakość klasyfikatora uśmiechu udostępnionego przez twórców biblioteki.

Zaskakujący może okazać się fakt, że biblioteka *OpenCV* w przypadku bardziej złożonego algorytmu sprawdziła się lepiej. Problem prawdopodobnie znajduje się po stronie klasyfikatora uśmiechu – dużo lepiej sprawdza się on dla mniejszych obrazów, po wycięciu samej twarzy.

8. PODSUMOWANIE

Ponieważ żadna z wykorzystanych bibliotek nie była w stanie wykryć uśmiechu, należy wykorzystać bardziej zaawansowany klasyfikator wykrywania uśmiechu lub wygenerować własny, na podstawie próbek negatywnych i pozytywnych. Jednocześnie wydaje się, że taki klasyfikator będzie wystarczający do detekcji uśmiechu – w przypadku oczu i nosa sprawdzają się one na wystarczająco wysokim poziomie.

Do zwiększenia dokładności prezentowanych wyników można wykorzystać wspomniany wcześniej algorytm wykrywania dolnej części twarzy. Jeżeli będzie on niewystarczający, można go rozbudować o kolejne warunki (np. „usta muszą znajdować się tuż pod nosem”).

Biblioteka *OpenIMAJ*, chociaż posiada dużo wygodniejsze API i wsparcie do programowania w języku *Java*, pokazuje znacznie mniejszą wydajność w stosunku do biblioteki *OpenCV*. W związku z tym spostrzeżeniem, jako główną bibliotekę do realizacji zadania zdecydowano się wykorzystać *OpenCV*, które wymaga dołączenia zewnętrznych bibliotek napisanych w C++, jednak dzięki temu oferuje dużo lepsze czasy detekcji.

9. LITERATURA

1. *Haar Cascades*, <http://alereimondo.no-ip.org/OpenCV/34> (data dostępu: 06.02.2017)
2. *OpenCV Haarcascades*,
<https://github.com/opencv/opencv/tree/master/data/haarcascades> (data dostępu:
06.02.2017)
3. *Smile detection with OpenCV, the „nose” trick*,
<https://aminesehili.wordpress.com/2015/09/20/smile-detection-with-opencv-the-nose-trick/> (data dostępu: 06.02.2017)