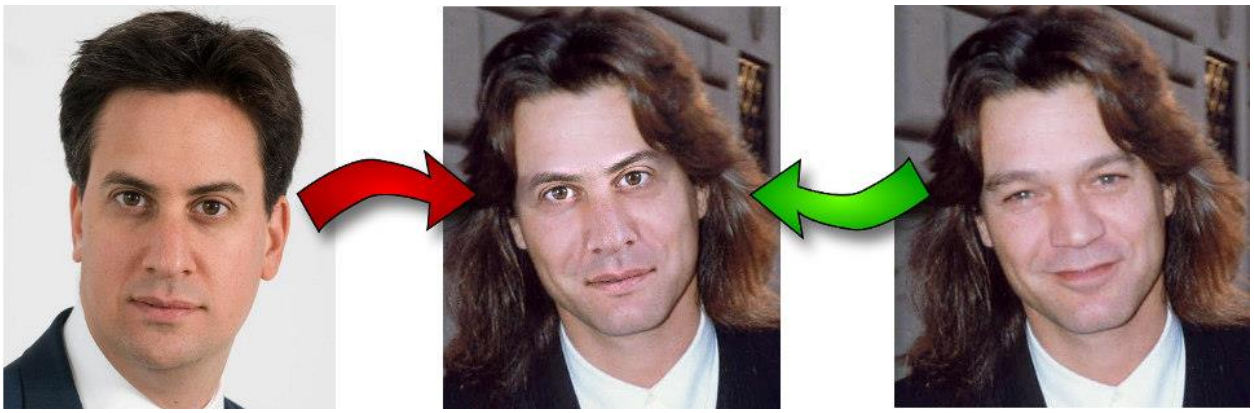
Проектна задача

по предметот Обработка на слика

Детекција на лице – замена на лица



Давид Ристов

Број на индекс: 151029

Професор

Димитровски д-р Ивица

1. Препознавање на лице со библиотеката dlib

1.1 Вовед

Главниот проблем при препознавање на лице е наоѓање на обележјата на лицето(facial landmarks). Обележјата на лицето се користат за да ги локализираат и претстават истакнатите региони од лицето како што се очите,веѓите,носот,устата и вилицата. Обележјата на лицето успешно се применуваат за решавање на следните проблеми:

- Face alignment
- Head pose estimation
- Face swapping
- Blink detection

1.2 Што се обележја на лицето(facial landmarks) ?

Детектирањето на обележја на лице е проблем од подмножеството на проблеми за детектирање на облик на лице(shape prediction problem). Ако е дадена влезна слика(и вообичаено почетен ROI – Region of interest која го одредува објектот од интерес), shape predictor-от се обидува да ги локализира клучните точки од интерес во однос на формата.

Во контекстот на обележја на лице, главната цел е да се откријат важните структури од лицето(facial structures) користејќи методи за предвидување на облик на лице. Затоа, откривањето на обележјата на лицето е процес кој се состои од два чекора:

- Локализација на лицето од влезната слика
- Откривање на главните facial structures знаејќи го лицето(face ROI)

Постојат различни детектори за обележја на лице, но сите методи во суштина се обидуваат да ги локализираат следните региони од лицето: уста, лева веѓа, десна веѓа, лево око, десно око, нос и вилица. Детекторот на обележја на лице којшто е вклучен во dlib библиотеката е имплементација на алгоритамот Ensemble of Regression Trees(Алгоритам којшто користи повеќе регресиски дрва за наоѓање на обележја на лице) опишан во весникот објавен од Kazemi and Sullivan (2014).

Детекторот којшто го користам во овој проект е HOG(Histogram of oriented gradients) + Linear SVM(Support Vector Machines) object detector.

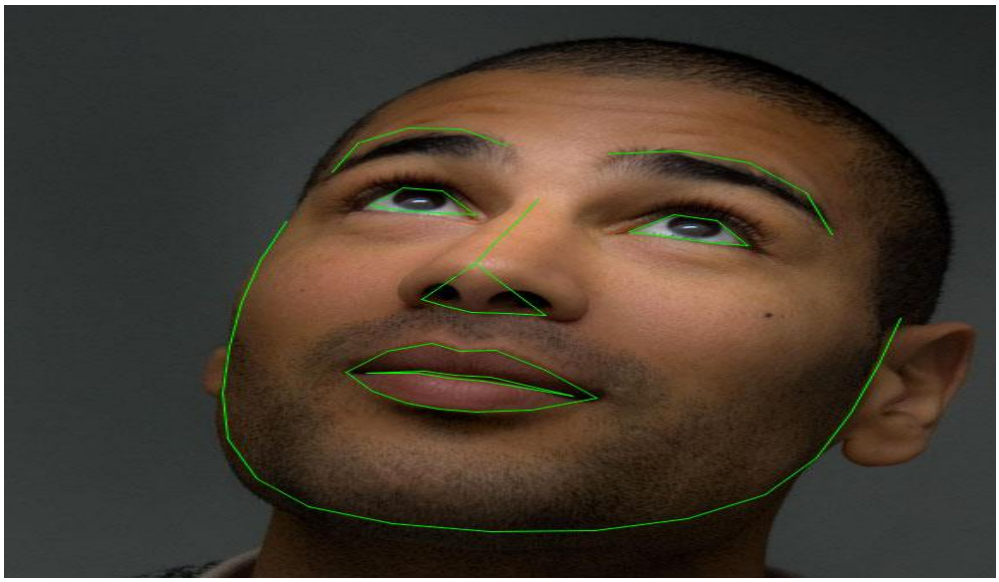


Figure 1: Facial landmarks are used to label and identify key facial attributes in an image

1.3 Разбирање на детекторот на обележја на лице од библиотеката dlib

Претходно обучениот детектор на обележја на лице во библиотеката dlib се користи за проценка на локацијата на 68 (x,y) – координати кои се мапираат во facial structures од лицето. Овој детектор е претходно обучен користејќи го iBUG-300 W податочното множество и како таков се користи готов во библиотеката dlib.

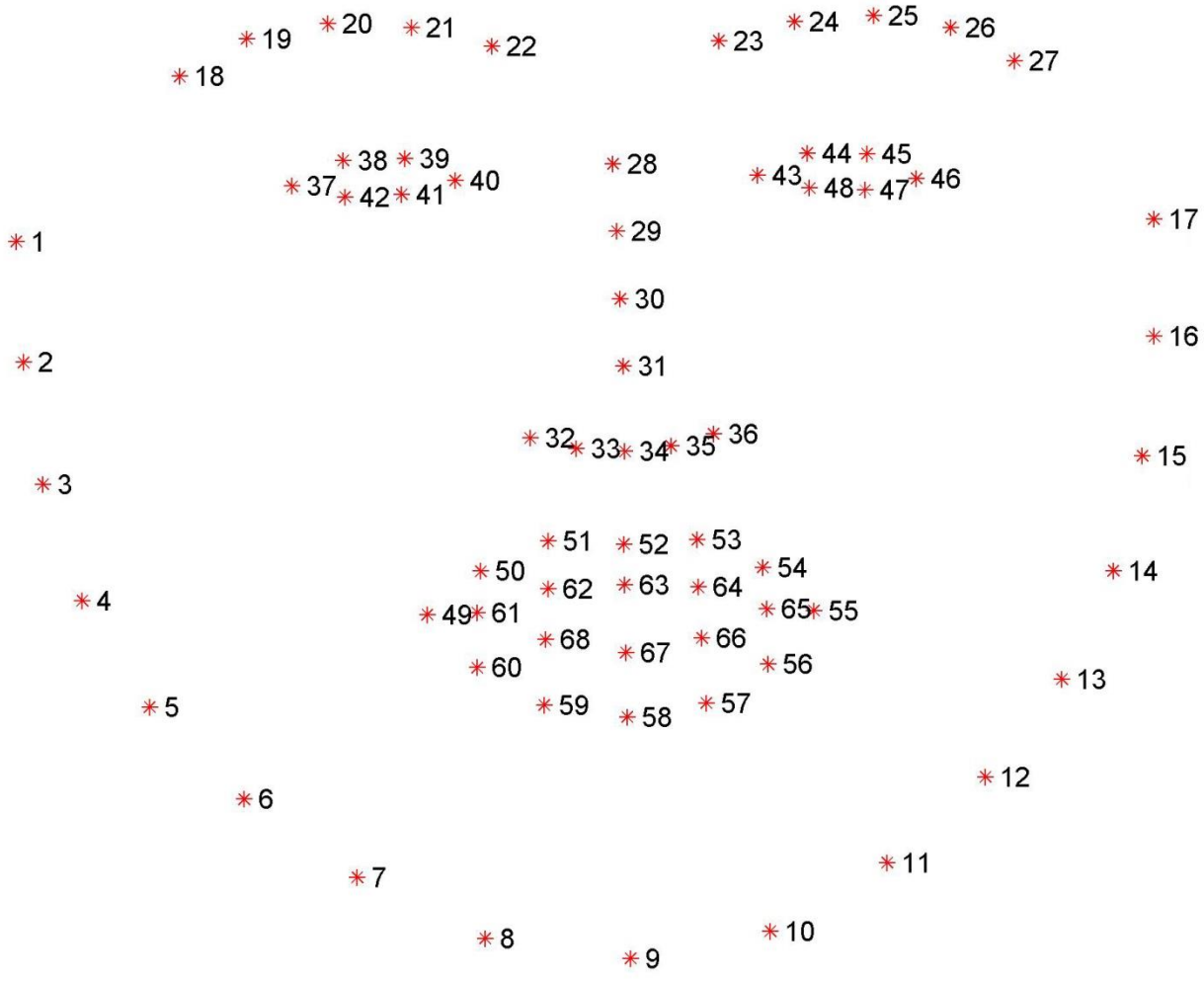


Figure 2: Visualizing the 68 facial landmark coordinates from the iBUG 300-W dataset

1.4 Ensemble of regression trees алгоритамот

Ќе покажеме дека проблемот на детекција на лице ефикасно може да се реши со секвенца од повеќе регресиски функции. Во нашиот случај, секоја регресиска функција од секвенцата регресиски функции ефикасно го проценува обликот од првичната проценка и интензитетот на ретко множество од пиксели кои биле индексирани во однос на оваа првична проценка.

Главно се среќаваат два клучни елементи во научените регресиски функции:

- Првиот елемент се врти околу индексирањето на интензитетот на пикселите во однос на моменталната проценка на обликот. Извлечените карактеристики(features) којшто имаат вектор репрезентација од дадена слика со лице може многу да се разликуваат поради различни фактори како што се деформација на обликот на лицето и условите на осветлување на самата слика. Затоа, проценката на обликот на лицето со користење на само регресиските функции е тешко. Дилемата е во тоа што ни требаат сигурни карактеристики коишто точно ќе го предвидуваат обликот на лицето, а од друга страна ни треба и точна проценка на обликот на лицето во следната итерација за пак да може да бидат извлечени што е можно поточни карактеристики за самиот облик. За решавање на овој проблем, се користи итеративен пристап така што наместо глобалниот координатен систем на сликата(целата слика), сликата се трансформира во нормализиран координатен систем користејќи **билинеарна интерполација** врз основа на тековната проценка на обликот

на лицето од тековната итерација и оваа трансформација се користи за да се извлечат карактеристиките и да се ажурира тековниот вектор од параметрите на обликот на лицето. Овој процес вообичаено се повторува неколку пати до конвергенција.

- Вториот елемент се однесува на проблемот со предвидување и естимација на карактеристики коишто најдобро би го опишале обликот на самото лице. Овој проблем, гледано математички е не-конвексен и има многу локални оптимуми. Алгоритамот се справува успешно со овој проблем така што претпоставува дека проценетата форма мора да лежи во линеарен подпростор којшто пак се наоѓа со користење на principal components(користејќи го алгоритамот PCA). Оваа претпоставка во голема мера го намалува бројот на потенцијални проценети форми и помага при избегнување на локалните оптимуми. Главната идеја е во тоа што секогаш постои секвенца од регресиски функции(регресори) коишто гарантираат предвидување на облици коишто ќе лежат во линеарниот подпростор.

1.4.1 Форма на регресорот и процесот на тренирање

1.4.1.1 Форма на регресорот

Нека $x_i \in R^2$ биде x, y – координатите на i-тиот facial landmark на дадена слика I. Тогаш векторот $S = (x_1^T, x_2^T, ..., x_p^T)^T \in R^{2p}$ ги означува координатите на сите p facial landmarks за сликата I. Нека $S^{(t)}$ ја означува моменталната проценка на векторот S. Секој

регресор $r_t(.,.)$ во секвенцата од регресиски функции го предвидува ажурираниот вектор од сликата I и $S^{(t)}$ кое се додава за да се подобри проценката:

$$S^{(t+1)} = S^{(t)} + r_t(I,S^{(t)}) \quad (1)$$

Главната идеја е дека регресорот r_t ги прави своите предвидувања базирани на карактеристики, како што е интензитетот на пикселите пресметани од сликата I и индексирани во однос на моменталната проценка на обликот $S^{(t)}$. Ова воведува некоја форма на геометриска инваријантност во процесот и како што се зголемува бројот на итерации, можеме да бидеме сигурни дека секогаш се индексира прецизна семантичка локација на лицето.

Излезот продуциран од секвенцата на регресиски функции секогаш ќе лежи во линеарниот подпростор од податочното множество за обука ако првичната проценка $S^{(0)}$ припаѓа во овој линеарен подпростор. Затоа, не треба да се воведуваат дополнителни ограничувања што во голема мера го поедноставува методот. Почетната форма може едноставно да биде избрана како среден облик(mean shape) од податоците за обука центрирани и скалирани според излезот од bounding box(пронаоѓање на лицето од слика) од некој генерички детектор на лице.

1.4.1.2 Процесот на тренирање со gradient tree boosting алгоритамот

Да претпоставиме дека имаме податоци за обука $(I_1,S_1),\ldots,(I_n,S_n)$ каде што I_i е сликата којашто го содржи лицето и S_i е векторот од обликот на лицето. За да се научи првата функција на регресија r_0 од секвенцата на регресиски функции, од множеството за обука креираме торка којашто ја содржи сликата, почетната проценка на сликата и целниот чекор на ажурирање, тоа е,

$(I_{\pi_i}, S_i^{(0)},\Delta S_i^{(0)})$ каде:

$$\pi_I \in \{1,\ldots,n\} \tag{2}$$

$$S_i^{(0)} \in \{S_1,\ldots,S_n\}\backslash S_\pi \tag{3}$$

$$\Delta S_i^{(0)} = S_{\pi_i} - S_i^{(0)} \tag{4}$$

За i = 1,...,N. Вкупниот број на вакви торки ќе биде N = nR, каде R е бројот на иницијализации од слика I_i. Од овие податоци се изучува функцијата на регресија r_0 со користење на gradient tree boosting алгоритамот(види подоле) со сума од квадрати функцијата на загуба. Множеството на обука од торките се ажурира за да во следната итерација претставува ново множество на обука $(I_{\pi_i}, S_i^{(1)},\Delta S_i^{(1)})$ за следниот регресор r_1 од секвенцата така што:

$$S_i^{(t+1)} = S_i^{(t)} + r_t(I_{\pi_i}, S_i^{(t)}) \tag{5}$$

$$\Delta S_i^{(t+1)} = S_{\pi_i} - S_i^{(t+1)} \tag{6}$$

Овој процес се повторува се додека не бидат изучени секвенца од T функции на регресија $r_0, r_1, \ldots r_{T-1}$ така што кога ќе се искомбинираат, ќе дадат задоволително ниво на точност при предвидувањето на обликот на лицето.

Алгоритам 1 Изучување на регресорот r_t од секвенцата регресори

Имајќи го множеството за обука $\{(I_{\pi_i}, S_i^{(t)}, \Delta S_i^{(t)})\}_{i=1}^N$ и чекорот на учење(таканаречен shrinkage factor) $0 < v < 1$

1. Иницијализирај

$$f_0(I, S^{(t)}) = \arg\min_{\gamma \in R^{2p}} \sum_{i=1}^{N} || S_i^{(t)} - \gamma ||^2$$

2. За k=1, . . . , K:

   (a) За i = 1, . . . , N

$$r_{ik} = \Delta S_i^{(t)} - f_{k-1}(I_{\pi_i}, S_i^{(t)})$$

   (b) Фитувај регресиско дрво на целните r_{ik} со што се добива слаба регресиска функција $g_k(I, S^{(t)})$.

   (c) Ажурирај

$$f_k(I, S^{(t)}) = f_{k-1}(I, S^{(t)}) + v\, g_k(I, S^{(t)})$$

3. Излез $r_t(I,S^{(t)}) = f_K(I, S^{(t)})$

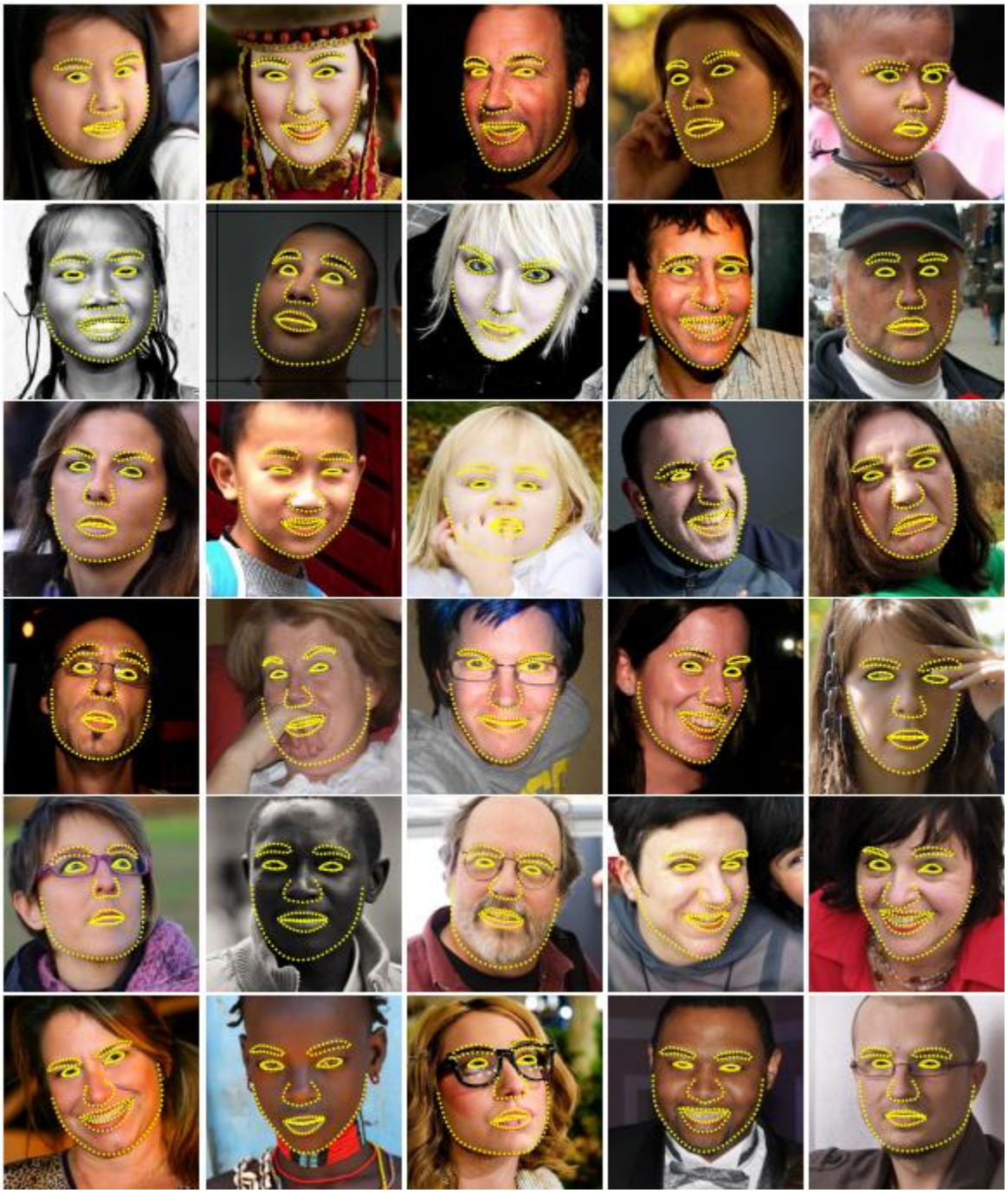


Figure 3 **Final results on the iBUG-300W database using 68 (x, y) coordinates as facial landmarks**

2. Bilinear interpolation(билинеарна интерполација)

2.1 Вовед

Во математиката, линеарна интерполација е метод на прилагодување на крива со користење на линеарни полиноми нови податочни точки во опсегот на дискретно множество од познати податочни точки.

Билинеарната интерполација претставува продолжување на линеарната интерполација за интерполирачки функции со две променливи на ректилинеарна дводимензионална решетка(rectilinear 2D grid). Главната идеја е да се изврши линеарна интерполација прво во една насока, а потоа повторно во другата насока. Иако секој чекор од билинеарната интерполација е линеарен во однос на примерочните вредности и нивната локација, интерполацијата во целост не е линеарна, туку квадратична во однос на локацијата на примероците.

2.2 Алгоритам

Да претпоставиме дека сакаме да ја најдеме вредноста на непознатата функција f во точката (x, y). Се претпоставува дека вредноста на f во четирите соседни точки се знае: $Q_{11} = (x_1,y_1)$,

$Q_{12} = (x_1, y_2)$, $Q_{21} = (x_2,y_1)$, $Q_{22} = (x_2,y_2)$.

Алгоритам 2 Билинеарна интерполација

1. Направи линеарна интерполација по x-оската

$$f(x, y_1) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})$$

$$f(x, y_2) \approx \frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})$$

2. Направи линеарна интерполација по y-оската

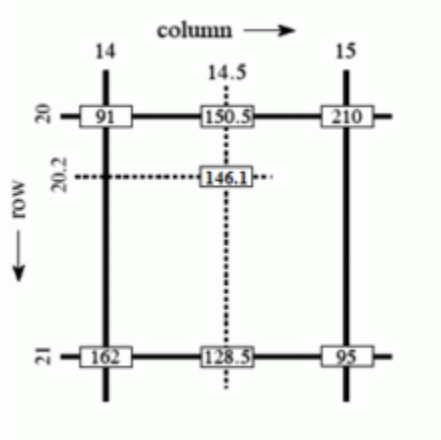
$$f(x, y) \approx \frac{y_2 - y}{y_2 - y_1} f(x, y_1) + \frac{y - y_1}{y_2 - y_1} f(x, y_2)$$

$$= \frac{y_2 - y}{y_2 - y_1} \left(\frac{x_2 - x}{x_2 - x_1} f(Q_{11}) + \frac{x - x_1}{x_2 - x_1} f(Q_{21})\right) + \frac{y - y_1}{y_2 - y_1}\left(\frac{x_2 - x}{x_2 - x_1} f(Q_{12}) + \frac{x - x_1}{x_2 - x_1} f(Q_{22})\right)$$

$$= \frac{1}{(x_2 - x_1)(y_2 - y_1)}\left(f(Q_{11})(x_2 - x)(y_2 - y) + f(Q_{21})(x - x_1)(y_2 - y) + f(Q_{12})(x_2 - x)(y - y_1) + f(Q_{22})(x - x_1)(y - y_1)\right)$$

$$= \frac{1}{(x_2 - x_1)(y_2 - y_1)} \begin{bmatrix} x_2 - x & x - x_1 \end{bmatrix} \begin{bmatrix} f(Q11) & f(Q12) \\ f(Q21) & f(Q22) \end{bmatrix} \begin{bmatrix} y_2 - y \\ y - y_1 \end{bmatrix}$$

Пример на билинеарна интерполација



$I_{20,14.5} = \frac{15-14.5}{15-14}$ x 91 + $\frac{14.5-14}{15-14}$ x 210 = 150.5

$I_{21,14.5} = \frac{15-14.5}{15-14}$ x 162 + $\frac{14.5-14}{15-14}$ x 95 = 128.5

$I_{20.2,14.5} = \frac{21-20.2}{21-20}$ x 150.5 + $\frac{20.2-20}{21-20}$ x 128.5 = 146.1

Алгоритми за детекција со голема прецизност

Алгоритмите за детекција и замена на лице главно се засноваат на понапредни техники за обработка на слика и генерално се уште не постои алгоритам кој ќе дава 100% точност во било кој случај и како такви се уште се подобруваат. Дел од понапредните техники се конволуциските невронски мрежи и генетските алгоритми, дрва на одлука и напредни SVM, кои се користат за учење на дадени

параметри од големи податочни множества и побаруваат специфични хардверски спецификации за да можат да бидат извршени на одредена машина. Ваквите понапредни техники излегуваат од рамките на овој курс.

Референци

1. Билинеарна интерполација

https://en.wikipedia.org/wiki/Bilinear_interpolation

2. Весникот на Каземи и Саливан

https://www.semanticscholar.org/paper/One-millisecond-face-alignment-with-an-ensemble-of-Kazemi-Sullivan/1824b1ccace464ba275ccc86619feaa89018c0ad

http://www.csc.kth.se/~vahidk/papers/KazemiCVPR14_poster.pdf

3. Блог

https://www.pyimagesearch.com/2018/04/02/faster-facial-landmark-detector-with-dlib/

4. Скалирање на слика

https://en.wikipedia.org/wiki/Image_scaling