УДК 00.007.3

Глибовець А. М., Збунь В. В.

ПІДСИСТЕМА ПОШУКУ ЗОБРАЖЕНЬ В ІНТЕРНЕТ

На сьогодні проблему індексації та пошуку текстової інформації майже вирішено, залишилися лише певні моменти покращення адаптованості алгоритмів до потреб користувача. Усі сучасні пошукові системи поставили собі за мету в найкоротші терміни розв'язати проблеми ефективного пошуку зображень та відеоматеріалів. Одній із цих проблем, а саме побудові підсистеми пошуку зображень, присв'ячено цю публікацію. У статті розглянуто основні алгоритми та наведено їх реалізацію.

Ключові слова: пошук зображень, content-based image retrieval, перцептивні хеш-алгоритми, метод колірних гістограм.

Сьогодні фотографії вважають одним із найпоширеніших і найпопулярніших видів відображення навколишнього. Легкість відтворення на матеріальному носії, передання в електронному форматі призводить до збільшення кількості випадків неправомірного використання фотографічних творів, а отже, й порушення авторських прав на них. Така ситуація зумовлює необхідність формування ефективної системи технічних і правових засобів захисту авторських прав на фотографічні твори.

Згідно зі статтею 2 Бернської конвенції про охорону літературних і художніх творів та статтею 8 Закону України «Про авторське право і суміжні права» фотографічні твори, у тому числі твори, виконані способами, подібними до фотографії, охороняються як об'єкти авторського права. Для захисту від порушень авторського права в мережі Інтернет рекомендують застосовувати технічні засоби захисту фотографій, наприклад, водяний знак з іменем автора або цифрову стенографію, що дасть змогу непомітно від людського ока записати інформацію у файл із зображенням.

Традиційно для пошуку зображень використовують їхні текстові характеристики: ім'я файлу, заголовок, ключові слова тощо. Однак для пошуку візуально подібних зображень застосовують саме пошук зображень за змістом.

Пошук зображень за змістом (Content-based image retrieval) – це одна з технік машинного зору, що вирішує проблему отримання зображень, що є проблемою пошуку цифрових зображень у великих базах даних. Пошук зображень за змістом є протилежним підходом до пошуку зображень за описом.

Сучасні CBIR-системи працюють у два етапи: індексування і пошук [1]. На етапі індексування кожний образ у базі даних представляється вектором властивостей. Універсальні системи CBIR відносять образ до однієї із трьох категорій залежно від підходу отримання його властивостей: гістограма, кольорове розташування і пошук за регіонами. Такими властивостями, зокрема, є колір, форма, структура і розташування. Отримані властивості зберігаються в окремій базі даних візуальних властивостей. На етапі пошуку обчислюють властивості із образу-запиту користувача. Використовуючи критерії подібності, отриманий вектор властивостей порівнюють з векторами у базі даних візуальних властивостей. Користувач у відповідь отримує образи, які максимально відповідають запиту.

У процесі пошуку зображень за візуальним змістом процес порівняння має здійснюватися не за вихідними зображеннями, а за даними, отриманими в результаті попередньої обробки цих зображень. У разі запису даних у строковому форматі, ефективність методу залежить від алгоритмів, використовуваних для формування рядка. Одним з найбільш ефективних і універсальних є використання перцептивного хеш-алгоритму.

Перцептивні хеш-алгоритми описують клас функцій для генерації порівнянних хешів. Вони використовують різні властивості зображення для побудови індивідуального «відбитка». Надалі ці «відбитки» можна порівнювати один з одним.

Алгоритми обчислення перцептивного хешу мають однакові базові властивості:

– зображення можна змінювати в розмірі;
– змінювати співвідношення сторін;

– змінювати колірні характеристики (контраст, колір, яскравість і т. д.).

Однак при зміні цих властивостей зображення все одно збігатимуться за хешем. Для отримання хешу є кілька поширених алгоритмів. Перцептивні хеші – це інша концепція порівняно з криптографічними хеш-функціями на зразок MD5 і SHA1. У криптографії кожен хеш є випадковим. Дані, які використовують для генерації хешу, виконують роль джерела випадкових чисел, тож однакові дані дадуть однаковий результат, а різні дані – різний результат. Із порівняння двох хешів SHA1 насправді можна зробити тільки два висновки. Якщо хеші відрізняються, отже, дані різні. Якщо хеші збігаються, то і дані, швидше за все, однакові (оскільки існує ймовірність колізій, то однакові хеші не гарантують збігу даних). На відміну від них, перцептивні хеші можна порівнювати між собою і робити висновок про ступінь відмінності двох наборів даних.

Існує багато різних підходів до формування перцептивного хешу зображення. Всіх їх об'єднують 3 основні етапи:

1. Попередня обробка.
2. Основні обчислення.
3. Побудова хешу.

Метод колірних гістограм

Ідея методу колірних гістограм для порівняння зображень зводиться до такого. Вся множина кольорів розбивається на набір непересічних підмножин, що повністю її покривають. Для зображення формують гістограму, що відображає частку кожної підмножини кольорів у колірній гамі зображення. Для порівняння гістограм вводиться поняття відстані між ними. Відомі різні способи побудови і порівняння колірних гістограм, що відрізняються між собою початковою колірною схемою, розмірністю гістограми і визначенням відстані між гістограмами.

При розбитті RGB-кольорів за яскравістю обчислюють інтенсивність кожного кольору на підставі його червоної, синьої і зеленої складових. Отримане значення, укладене між числами 0 і 255, потрапляє в один зі 16 інтервалів, на які розбивається діапазон можливих значень. Як відстань між гістограмами використовують суму модулів різниці відповідних елементів гістограм; деяке удосконалення методу досягається при обчисленні відстані на підставі поелементного порівняння гістограм з урахуванням сусідніх елементів. Цей метод найбільш ефективний для чорно-білих напівтонових зображень. Для кольорових RGB-зображень кращі результати дає інший спосіб – розбиття RGB-кольорів за прямокутними паралелепіпедами.

Кольоровий RGB-простір розглядається як тривимірний куб, кожна вісь якого відповідає одному з трьох основних кольорів (червоному, зеленому або синьому), ділення на осях пронумеровані від 0 до 255 (більше значення відповідає більшій інтенсивності кольору). За такого розгляду будь-який колір RGB-зображення може бути представлений точкою куба. Для побудови колірної гістограми кожна сторона ділиться на 4 рівні інтервали, відповідно RGB-куб ділиться на 64 прямокутні паралелепіпеди.

Гістограма зображення відображає розподіл точок RGB-простору, відповідних кольорам пікселів зображення. Як відстань між гістограмами використовують покомпонентну суму модулів різниці між ними. Незважаючи на граничну простоту підходу, він показує досить стабільні результати. Наприклад, якщо необхідна картинка в яскраво-червоних тонах, можна взяти за зразок фотографію червоної троянди і отримати з бази як результат пошуку фотографії, на яких великим планом зображено червоний прапор, черепичний дах, жінку в червоній сукні і т. д.

Більш точне порівняння зображень досягається за допомогою техніки квадродерева, коли методи обчислення і порівняння колірних гістограм застосовують не до всього зображення, а до його чверті (однієї шістнадцятої і т. д.).

У методі колірних гістограм порівняння зображень ґрунтується на Евклідовій відстані у просторі відстаней між гістограмами їхніх частин. Цей метод дає результат, семантично відмінний від інших варіантів: зображення, що розрізняються тільки взаємним розташуванням ідентичних за кольором об'єктів, вважають несхожими, однак їх можна було визначити як близькі без використання цієї техніки. Але при цьому однакові зображення різного кольору, або з перефарбованою в інший колір ділянкою, будуть розпізнані як різні, тому потрібно обрати перцептивний-хеш алгоритм.

Метод перцептивних хешів

Існує 4 алгоритми перцептивних хешів [3]:

1. Block Mean Value Based Hash.

Суть алгоритму полягає у відображенні середнього значення низьких частот. У зображеннях високі частоти забезпечують деталізацію, а низькі – показують структуру. Тому для побудови такої хеш-функції, яка для схожих зображень видаватиме близький хеш, потрібно позбутися високих частот. Отриманий хеш стійкий до

масштабування, стискання або розтягування зображення, зміни яскравості, контрасту, маніпуляціями з кольорами. Але головна перевага алгоритму – швидкість роботи. Для порівняння хешів цього типу використовують функцію нормованої відстані Хеммінга.

2. DCT Based Hash.

Дискретне косинусне перетворення (ДКП, DCT) – одне з ортогональних перетворень, тісно пов'язане з дискретним перетворенням Фур'є (ДПФ) і є гомоморфізмом його векторного простору. ДКП, як і будь-яке Фур'є-орієнтоване перетворення, висловлює функцію або сигнал (послідовність із кінцевого числа точок даних) у вигляді суми синусоїд із різними частотами і амплітудами. ДКП використовує тільки косинусні функції, на відміну від ДПФ, що використовує і косинусні, і синусні функції.

Головними перевагами такого хешу є стійкість до малих поворотів, розмиття і стиснення зображення, а також швидкість порівняння хешів, завдяки їхньому маленькому розміру. Для порівняння хешів цього типу також використовують функцію відстані Хеммінга.

3. Radial Variance Based Hash.

Ідея алгоритму полягає в побудові плечового вектора дисперсії (ПВД) на основі перетворення Радона. Потім до ПВД застосовують ДКП і обчислюють хеш. Перетворення Радона – це інтегральне перетворення функції багатьох змінних вздовж прямої. Воно стійке до обробки зображень за допомогою різних маніпуляцій (наприклад, стиснення) і геометричних перетворень (наприклад, поворотів). Для порівняння хешів цього типу використовують пошук піку взаємнокореляційної функції.

4. Marr-Hildreth Operator Based Hash.

Оператор Марра-Хілдрет дає змогу визначати краї на зображенні. Їх можна визначити як край або контур, що відокремлює сусідні частини зображення, які мають порівняно відмінні характеристики. Цими характеристиками можуть бути колір або текстура, але частіше за все використовують сіру градацію кольору зображення (яскравість). Результатом визначення меж є карта кордонів, що описує класифікацію меж для кожного пікселя зображення. Якщо межі визначати як різку зміну яскравості, то для їх знаходження можна використовувати похідні або градієнт.

Порівняно з алгоритмом Radial Variance Based Hash, порівняння двох хешів займає досить незначний час, хоч розмір хешів відносно великий, оскільки використовується функція нормованої відстані Хеммінга. Цей алгоритм чутливий до поворотів зображення, але стійкий до масштабування, затемнення, стиснення.

У таблиці подано офіційні результати тестування швидкості роботи алгоритмів із книги «Implementation and Benchmarking of Perceptual Image Hash Functions» [2].

Показник	DCT	MH	Radial	BMB
Загальний час (у секундах)	911	343	118	**58**
Середній час на зображення (у секундах)	9.7	3.6	1.3	**0.6**

Як видно з результатів тестування, найшвидшим виявився алгоритм Block Mean Value Based Hash. Результати тестування підтвердили очікування, і тому для порівняння зображень буде застосовано саме цей алгоритм.

Нижче наведено кроки алгоритму Block Mean Value Based Hash та їх реалізацію мовою PHP:

1. Зменшити розмір зображення до розміру 16×16 пікселів.

2. Прибрати колір. Маленьке зображення переводиться в градації сірого, тому хеш зменшується втричі.

Ці два кроки реалізовані так:

```
    list($width, $height) = getimagesize($imageFilePath);
    $img = $this->createImage($imageFilePath);
    $size = 16;
    $new_img = imagecreatetruecolor($size, $size);
    imagecopyresampled($new_img, $img, 0, 0, 0, 0, $size,
$size, $width, $height);
    imagefilter($new_img, IMG_FILTER_GRAYSCALE);

  private function createImage($imagePath) {
    $check = getimagesize($imagePath);

    if ($check[«mime»] == «image/jpeg») {
    return imagecreatefromjpeg($imagePath);
    }
    else if ($check[«mime»] == «image/png») {
       return imagecreatefrompng($imagePath);

    }
    else if ($check[«mime»] == «image/gif») {
       return imagecreatefromgif($imagePath);
    }
    else {
       return 0;
    }
  }
```

3. Знайти середнє. Обчислити середнє значення кольору для всіх 256 пікселів.

```
    $colors = array();
    $sum = 0;

       for ($i = 0; $i < $size; $i++) {
          for ($j = 0; $j < $size; $j++) {
             $color = imagecolorat($new_img, $i, $j) & 0xff;
             $sum += $color;
             $colors[] = $color;
          }
       }
       $avg = $sum / ($size * $size);
```

4. Побудувати ланцюжок бітів. Для кожного пікселя робиться заміна кольору на 1 або 0 залежно від того, більший він або менший ніж середній.

5. Побудувати хеш. Перевести 256 бітів в одне значення.

Останні 2 кроки реалізовано таким чином:

```
$hash = '';
$bits = '';
$count = 0;
foreach ($colors as $color) {
    if ($color > $avg) {
        $bits .= '1';
    } else {
    $bits .= '0';
 }
 $count++;
 if (!($count % 4) ) {
     $hash .= dechex(bindec($bits));
     $bits = '';
  }
 }
 return $hash;
```

Отриманий хеш стійкий до масштабування, стискання або розтягування зображення, зміни яскравості, контрасту, маніпуляцій із кольорами. Але головна перевага алгоритму – швидкість роботи. Для порівняння хешів цього типу використовують функцію нормованої відстані Хеммінга.

```
public function hammingDistance($hash1, $hash2) {
    $h1 = str_split($hash1);
    $h2 = str_split($hash2);
    $hd = 0;
    if( count($h1) == count($h2) ) {
       for ($i = 0; $i < count($h1); $i++)
          if($h1[$i] != $h2[$i]) $hd++;
       return $hd;
    }
    else {
       return 1;
    }
}
```

Опис принципу роботи пошукового робота

Спочатку користувач обирає зображення для пошуку та вказує посилання на WEB-сторінки для аналізу. Під час завантаження перевіряють відповідність MIME-типу, для підтвердження, що завантажений файл справді є зображенням. У разі якщо зображення пройшло перевірку, його порівнюють з усіма зображеннями з БД для уникнення зберігання копій. Якщо в базі було знайдене ідентичне, то останнє завантажене видаляють і далі використовують уже наявний ідентифікатор зображення.

Після цього по черзі для кажної із вказаних WEB-сторінок проводять парсинг на наявність посилань і зображень. Завантажене зображення порівнюють з усіма знайденими на сторінці зображеннями і у разі схожості зберігають посилання на поточну сторінку, на якій було виявлено збіг. Потім зберігають усі унікальні посилання, знайдені на поточній сторінці. Залежно від заданих параметрів парсингу, зберігають або усі посилання, або лише внутрішні.

Після аналізу усіх вказаних користувачем сторінок аналізуватимуться усі знайдені посилання в порядку їх знаходження, поки в базі не залишиться непроаналізованих посилань для заданого зображення.

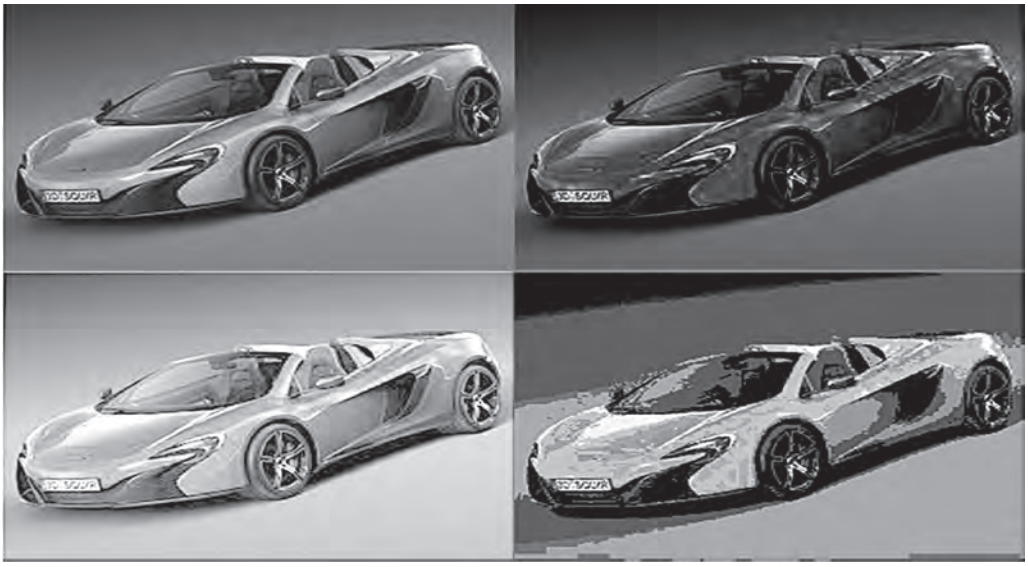
Для аналізу швидкості роботи реалізованого алгоритму було проведене тестування. Було відібрано 100 різнотипних зображень зі спільними особливостями. Загальний час обробки 100 зображень склав 39,52 секунди, це в середньому 0,39 секунди розрахунку хешу для одного зображення. Середній розмір зображень 0,13820579 мегабайта.

Портрети, зображені на нижньому рисунку, відповідно до реалізованого алгоритму є схожими, попри незначні відмінності.

Проте наступна пара зображень не була визнана як схожа, через значні відмінності у співвідношенні сторін зображень. Отже, алгоритм чутливий до зображень, що мають різні розміри.

Через дуже схожу форму букети, зображені нижче, були прийняті за схожі.

Чотири автомобілі, зображені нижче, підтверджують, що алгоритм не чутливий до зміни кольору, знебарвлення, погіршення якості та деяких ефектів, у цьому випадку постеризації.



Також незначні пошкодження та, наприклад, водяні знаки не завадять розпізнанню схожих зображень.

Висновки

Наведений прототип системи можна застосовувати для пошуку посилань на сторінки, на яких розташовані подібні або ідентичні фотографії, відслідковування зображення, що охороняються як об'єкти авторського права, чи просто пошуку посилання на WEB-сторінку на певному веб-сайті, з якого було збережено зображення, але загублено посилання на статтю.

До перспектив покращення можна віднести вдосконалення пошуку зображень по базі даних, а саме групування зображення за певними ознаками, наприклад за кольором. Також за наявності високої продуктивності ЕОМ можлива оптимізація багатопоточних обчислень для прискорення порівняння зображень, а тим самим і пошуку. Також можна внести зміну до алгоритму для повного аналізу зображення за будь-якого кута повороту або дзеркального відображення.

Список літератури

1. Огневой Г. Д. Методы и алгоритмы поиска изображений [Электронный ресурс] / Г. Д. Огневой. – Режим доступу: www.bntu.by/news/G7-conference-mido. – Назва з екрана.
2. Christoph Zauner. Implementation and Benchmarking of Perceptual Image Hash Functions / Christoph Zauner. – 2010.
3. Nick Pears. 3D Imaging, Analysis and Applications / Nick Pears, Yonghuai Liu, Peter Bunting. – 2012. – 497 p.

A. Glybovets, V. Zbun

SUBSYSTEM OF IMAGE RETRIAVAL FROM INTERNET

Presently the problem of indexing and searching textual information is almost resolved, leaving only some improvement of algorithms adaptability to the needs of the user. All modern search engines set a goal as soon as possible to solve the problem of effective image and video materials search. Regarding of these problems, a special building subsystem of image search is described in this article. This publication discusses the main algorithms and their implementation. The article describes the main content-based image retrieval algorithms such as perceptual hash algorithms and the method of color histograms. The article shows how these algorithms can be used for subsystem of image retrieval from the Internet and finding non-legal usages of images. The article contains algorithms, program realization, and conclusions. It is only the first part of work, and in the future works the authors plan to provide a fully functional system that will work on-line.

Keywords: image retrieval, content-based image retrieval, perceptual hash algorithms, method of color histograms

Матеріал надійшов 20.09.2016