КОМП'ЮТЕРНІ ЗАСОБИ, МЕРЕЖІ ТА СИСТЕМИ

УДК 004.4

О.В. БАГАЦЬКИЙ

ІНСТРУМЕНТАЛЬНИЙ КОМПЛЕКС ДЛЯ ДОСЛІДЖЕННЯ АЛГОРИТМІВ ВИДІЛЕННЯ КОНТУРІВ ОБ'ЄКТІВ

A.V. Bagatskij

THE INSTRUMENTAL COMPLEX FOR RESEARCH ALGORITMUS FOR OBJECT BOUNDARIES

This article describes a program created to calculate the contours of objects, which allows the use OpenCV libraries and other libraries to detect object boundaries.
Key words: object boundaries, OpenCV, spatial filtering.

В данной статье описано созданную программу для расчета контуров объектов, позволяющую как использование библиотеки OpenCV, так и использование других библиотек для распознавания контуров объектов.
Ключевые слова: распознавание контуров, OpenCV, пространственная фильтрация.

В даній статті описано створену програму для обрахунку контурів об'єктів, яка дозволяє як використання бібліотеки OpenCV, так і використання інших бібліотек для розпізнавання контурів об'єктів.
Ключові слова: розпізнавання контурів, OpenCV, просторова фільтрація.

Постановка проблеми. В даний час все більшого розповсюдження набувають роботизовані системи, складовою яких є підсистеми штучного зору. Такі підсистеми використовуються для визначення контурів об'єктів, однак для такого визначення існує багато методів з різною ефективністю в залежності від освітлення, розмірів і положення у просторі об'єктів та ін. Крім того, необхідно мінімізувати розмір програми для подальшого використання створених алгоритмів у реальних пристроях.

Аналіз останніх досліджень та публікацій. Існує програмна бібліотека OpenCV [1], яка сумісна з мовами програмування C++ та Java і використовує подібні алгоритми для визначення контурів об'єктів, однак використання цієї бібліотеки значно збільшить розміри самої програми. З іншого боку, використання OpenCV надає більші можливості для обробки зображення, тому доцільно розробляти програму з можливістю використання цієї бібліотеки, оскільки для бібліотеки OpenCV все одно необхідно створювати допоміжний програмний код (інтерфейс), який би дозволяв завантаження та створення нового рисунку для проведення над ним операцій для виділення контуру і збереження результату у різних графічних форматах. Таким чином, створювані програмні засоби повинні бути сумісні з OpenCV та мати мінімально можливий розмір програми для виділення контуру.

Виділення невирішених раніше частин загальної проблеми, яким присвячується

стаття. На відміну від OpenCV demonstrator (GUI) [2] розроблена програма має більше інструментів для створення або корекції зображення. Також розроблена програма дозволяє використання сторонніх бібліотек для розширення можливостей обробки зображень; включає у себе алгоритми завантаження та обробки оригінального зображення, а також зберігання або друк результуючого зображення після використання методу виділення контуру. Інтерфейс користувача дозволяє налаштовувати програму та вибирати метод виділення контуру для зображення.

Методи виділення контуру зображення, які використовуються у даній програмі. У програмі використовуються відомі методи виділення контурів на зображенні за допомогою просторової фільтрації [3]. Перетворення базуються на аналізі зображення за допомогою обрахування яскравості кожного пікселя зображення використовуючи першу похідну (наприклад, перетворення оператором Собеля) або другу похідну (оператором Лапласа). В загальному випадку, результуюче зображення пікселя після використання оператора буде дорівнювати [3]:

$$R = \sum_{i=1}^{mn} w_i \cdot z_i, \tag{1}$$

де w_i – відповідний коефіцієнт вибраного оператора, z_i – значення відповідних пікселів зображення, mn – загальна кількість коефіцієнтів у оператора.

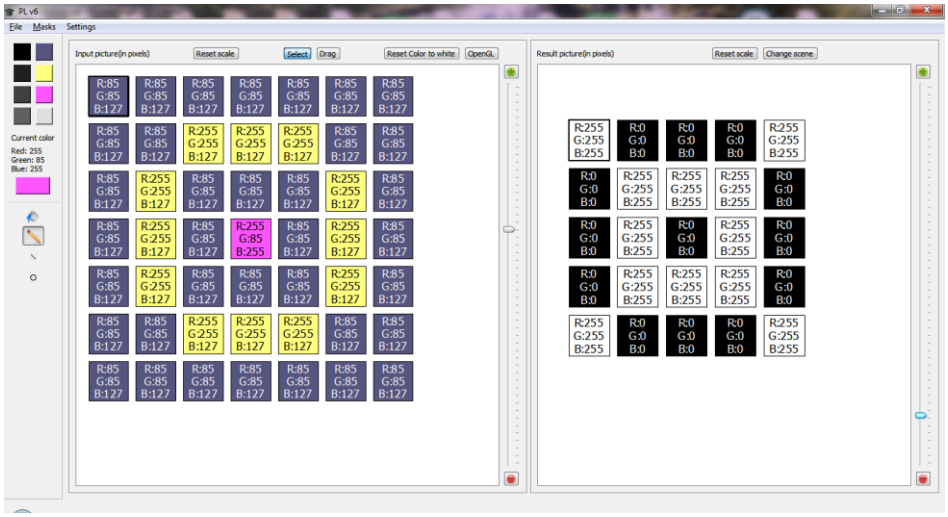
Результуюче зображення буде складатись із відповідних значень R з формули (1) і буде менше оригінального зображення по периметру на результат ділення націло розміру оператора на два.

Розроблена програма дозволяє створювати будь-які оператори для перетворення зображення, використовуючи будь-який розмір та коефіцієнти. Всі створені оператори зберігаються і можуть бути використані для подальших розрахунків.

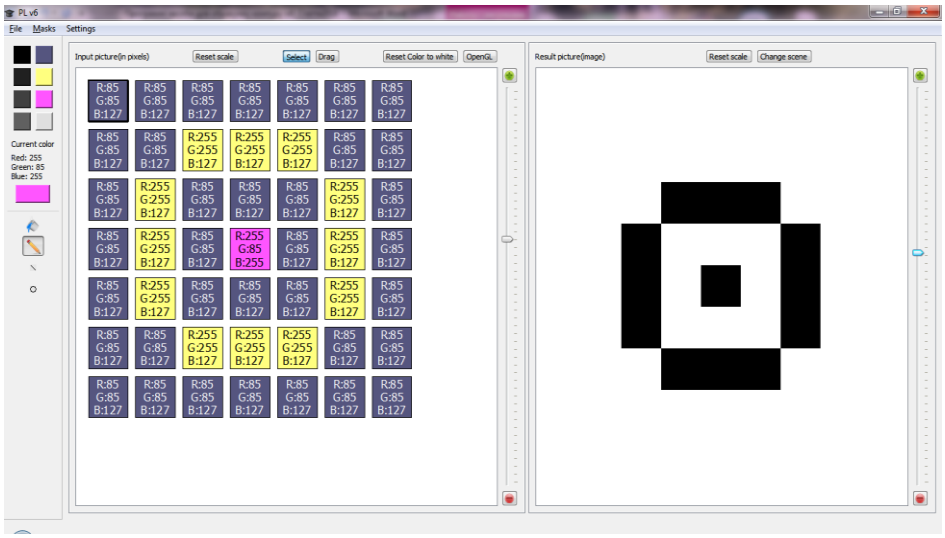
На рисунку показано інтерфейс користувача розробленої програми, який розроблявся за допомогою мови Qt. Вікно програми розділено на чотири області зверху знаходяться меню, під яким (зліва направо) – панель палітри кольорів та інструментів, вікно нового зображення, вікно результуючого зображення.

За допомогою меню користувач може створювати нове або завантажувати вже існуюче кольорове зображення для його подальшої обробки, друкувати результуюче зображення, вибрати або створити оператор просторової фільтрації, налаштувати програму.

Нове зображення з’являється у лівому вікні програми і буде складатися з графічних елементів-«пікселів», з якими будуть проводитися подальші маніпуляції. Кожен з елементів-«пікселів» при наближенні відображає свій колір у вигляді тексту в декілька рядків, причому цифри після літер «R», «G» та «B» є відповідними складовими червоного («R»), зеленого («G») та блакитного («B») кольорів.



а



б

РИСУНОК. Інтерфейс користувача програми для виділення контуру: а – перетворення «піксель» – «піксель», б – перетворення «піксель» – «рисунок»

Користувачу надається можливість вибору кольору за допомогою палітри кольорів, яка знаходиться з лівого краю вікна програми. За умовчуванням, палітра складається з восьми елементів, з кольорами у просторі RGB від (0, 0, 0) до (224, 224, 224) («відтінки сірого»). Залежність кількості елементів палітри та RGB-складової кольорів, які знаходяться у палітрі розраховується за допомогою формули:

$$C = \frac{256}{n} \cdot k,$$

де C – один з RGB-кольорів, n – загальна кількість елементів (доступних кольорів) палітри, k – поточний номер кольору.

Поточний колір «відтінка сірого» у просторі RGB з координатами (C_R, C_G, C_B) буде визначатися як $C_R = C_G = C_B = C$.

Існує можливість вибору будь-якого кольору з RGB простору для певної позиції кольору в палітрі – необхідно виконати «подвійний клік» на елемент для відкриття вікна вибору кольору. Змінений таким чином колір елемента палітри зберігається при завершенні програми та автоматично відновлюється при наступному старті.

Для інформування в палітрі знаходиться елемент відображення поточного кольору, що використовується для створення зображення. Індикація виконується як у вигляді тексту, який інформує про частку RGB-кольорів у поточному кольорі, так і у вигляді відображення кольору.

Під палітрою кольорів знаходиться панель інструментів, за допомогою якої можна створювати або коригувати існуюче зображення. Вона складається з чотирьох елементів – «заливка» («floodfill»), «точка» («point»), «лінія» («line») та «коло» («circle»). Зміна кольору відбувається при натисненні на вибраний «піксель».

Реалізація алгоритму «заливки» виконана за допомогою рекурсивного алгоритму заливки лініями («Recursive Scanline Floodfill Algorithm»), який детально описаний у [4]. Кольором, що заливається, є колір натиснутого графічного елемента-«пікселя»; колір, яким заливається, є поточний колір, який вибраний з палітри кольорів. Алгоритм використовує околицю фон Неймана («von Neumann neighborhood») [5] для вибору елементів-«пікселей», які повинні бути залитими.

Для зміни кольору певного графічного елемента-«пікселя» використовується елемент «точка», який є інструментом «по умовчанню» для режиму «створення/корекції» з кольором (0,0,0). При використанні цього елемента натиснутий «піксель» змінює свій колір на вибраний користувачем із палітри.

Для двох останніх елементів – «лінії» та «кола» – використовуються відповідні алгоритми Брезенхема («Bresenham») [6] для конструювання відображення лінії та кола. Ключовим елементом для прийняття рішення про зміну кольору «пікселя» у цьому алгоритмі є розрахунок похибки на кожному кроці алгоритму, який виникає при апроксимації лінії «пікселями» та порівняння розрахованої похибки з похибками координат. Для відображення лінії користувач по черзі натискає «пікселі», які мають стати початковою та кінцевою точками лінії на новому зображенні; для кола лінія, яка утворюється при натисканні «пікселів», є його радіусом.

Формула, яка використовується для розрахунку початкової похибки при апроксимації лінії з координатами початку (x_n,y_n) та кінця ($x_к$,$y_к$) за алгоритмом Брезенхема у цій програмі є [6]:

$$\delta_л = 2 \cdot (|x_к - x_п| + |y_к - y_п|),$$

де $\delta_л$ – значення похибки при апроксимації лінії, $(x_п,y_п)$ та $(x_к,y_к)$ – значення відповідних координат точок початку та кінця.

Для обрахування початкової похибки апроксимації кола у програмі використовується інша формула, а саме [6]:

$$\delta_к = 3 - 2 \cdot \sqrt{(x_к - x_п)^2 + (y_к - y_п)^2},$$

де $\delta_к$ – значення похибки при апроксимації кола, $(x_п,y_п)$ та $(x_к,y_к)$ – значення відповідних координат точок початку та кінця.

Праве вікно відображає результат взаємодії оригінального зображення з вибраним оператором. Результат відображається як у вигляді графічних елементів-«пікселів», так і у вигляді результуючого зображення. Відображення результату перемикається за допомогою кнопки, яка знаходиться над результуючим об’єктом. На відміну від нового зображення, результуюче зображення не можна змінювати, однак його можна зберегти у форматах зображень .png, .bmp та .xpm, які не спотворюють кольори зображення у кодуванні кольорів RGB.

Висновки.

1. Розроблені програмні засоби дозволяють створення або корекцію зображення для пошуку контуру об’єктів.

2. Розроблені програмні засоби мають можливість дослідження створених алгоритмів для розпізнавання контуру об’єкта та зменшення розміру програми для розпізнавання контурів об’єктів, що дозволить провадити обробку зображення на технічних засобах меншої складності.

1. http://opencv.org/
2. http://opencv.org/opencv-demonstrator-gui.html
3. *Гонсалес Р., Вудс Р.* Цифровая обработка изображений. М.: Техносфера, 2005. 1072 с.
4. http://lodev.org/cgtutor/floodfill.html
5. http://cell-auto.com/neighbourhood/vn/index.html
6. *Шилдт Г.* "Си" для профессиональных программистов. М., 1989.

Одержано 29.09.2016