УДК 629.7.051.83

СИСТЕМА ТЕХНІЧНОГО ЗОРУ В ЗАДАЧАХ СТАБІЛІЗАЦІЇ БЕЗПІЛОТНОГО ЛІТАЛЬНОГО АППАРАТУ

Котвицький Р.С., Сарибога Г.В.
Національний технічний університет України
«Київський політехнічний інститут»

Робота присвячена розробці системи технічного зору з метою стабілізації та керування безпілотним літальним апаратом. Наведено алгоритми обробки зображення. Розроблено загальну архітектура програмного забезпечення для реалізації системи, а також описана типова задача для технічного зору.
Ключові терміни: мікоконтролер, система технічного зору, розпізнавання зображень.

Постановка проблеми. Основною метою є розробка нової системи автоматичного керування безпілотного літального апарату (БПЛА) за допомогою системи технічного зору (СТЗ), а саме його стабілізація та пересування за об'єктом при виявленні цілі.

З метою вирішення задачі автоматичного керування БПЛА використовуються системи технічного зору. Системи технічного зору застосовуються для обробки і розпізнавання різних зображень, отриманих з телекамери або фотоапарата та є засобом спостереження і автоматичного прийняття рішення в тих чи інших випадках.

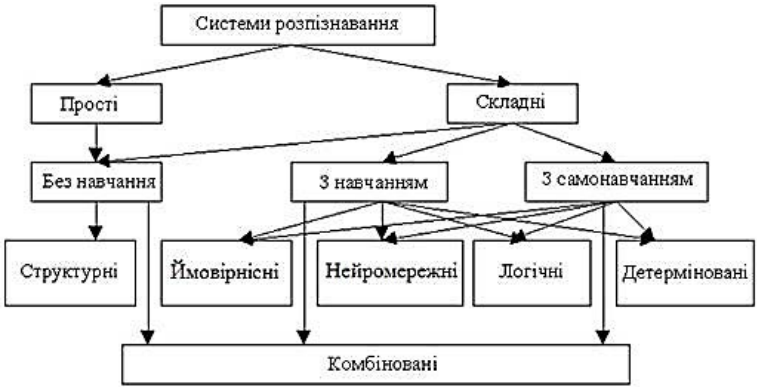

Рис. 1. Класифікація систем розпізнавання

Системи керування сучасними літальними апаратами (ЛА) призначені для керування складними багатофункціональними об'єктами, чинними в складній навколишній обстановці. При цьому канал зорового сприйняття є одним з найбільш важливих джерел інформації як у автоматичних, так і автоматизованих (людино-машинних) системах управління. Внаслідок цього в останні роки на передній план все більшою мірою виходять задачі створення систем технічного зору (СТЗ) для різних типів літальних апаратів (ЛА).

Аналіз останніх досліджень і публікацій. Розглянувши існуючи алгоритми розпізнавання зображень можна прийти до висновку щодо вибору методу розпізнавання в окремо взятому випадку.

СТЗ в даний час є одним з головних засобів розвитку автоматичних систем управління рухом в умовах, коли обсяг апріорної інформації не достатній і для вирішення завдань керування необхідно аналізувати навколишнє середовище в режимі реального часу.

Наразі дуже стрімко йде розвиток систем автоматичного керування на базі СТЗ, оскільки цей напрямок в сфері авіації є порівняно новим і має дуже великі можливості для подальшого використання.

Замість терміна «розпізнавання» часто використовують термін – «класифікація» (рис. 1). Постановки задач класифікації:

1. Задача ідентифікації, яка полягає у тому, щоб вирізнити певний конкретний об'єкт серед йому подібних.

2. Віднесення об'єкта до того чи іншого класу. Це може бути, наприклад, задача розпізнавання літер або прийняття рішення про наявність дефекту в деякій технічній деталі.

3. Кластерний аналіз, який полягає в розділенні заданого набору об'єктів на класи – групи об'єктів, схожі між собою за тим чи іншим критерієм. Цю задачу часто називають класифікацією без учителя, оскільки, на відміну від задачі 2, класи апріорно не задані.

Для дискримінантного розпізнавання, або для розпізнавання в просторі ознак, характерним є те, що кожен об'єкт зображується окремою точкою в деякому просторі. Координатними осями

```
import cv2
import numpy as np
import os
#########################################################################################
def main():

 capWebcam = cv2.VideoCapture(0)   #початок  захвату відеозображення  об'єкту та зв'язок з веб-
камерою
 print «defaultresolution = « + str(capWebcam.get(cv2.CAP  PROP_FRAME_WIDTH)) + «x» +
str(capWebcam.get(cv2.CAP_PROP_FRAME_HEIGHT))     #відображення оригінального розширення
камери
   capWebcam.set(cv2.CAP  PROP  FRAME  WIDTH, 320.0)    #трансформація розширення камери
   capWebcam.set(cv2.CAP_PROP_FRAME_HEIGHT, 240.0)  #зменшення значення розширення камери
для кращої швидкодії
   if capWebcam.isOpened() == False:                    #перевірка підключення камери
      print «error: cap Webcam not accessed successfully\n\n»
      os.system(«pause»)
      return
   # endif

  while cv2.waitKey(1) != 27 andcapWebcam.isOpened():          #програма працює доти, доки  не
  blnFrameReadSuccessfully, imgOriginal = capWebcam.read()  #нажати кнопку ESC або не вимкнути
камеру
      if not blnFrameReadSuccessfully or imgOriginal is None:    #перевірка на наявність кадру з камери
         print «error: frame not read from webcam\n»
                os.system(«pause»)
         break
      # endif
      #трансформація картинки з BGR в HSV простір
      imgHSV = cv2.cvtColor(imgOriginal, cv2.COLOR_BGR2HSV)
      #встановлення діапазону заданого кольору на зображені
      imgThreshLow = cv2.inRange(imgHSV, np.array([0, 170, 120]), np.array([20, 240, 255]))
      imgThreshHigh = cv2.inRange(imgHSV, np.array([20, 70, 170]), np.array([40, 170, 255]))

      imgThresh = cv2.add(imgThreshLow, imgThreshHigh)
      #створення маски зображення заданого кольору
      imgThresh = cv2.cvtColor(imgThresh, cv2.COLOR_GRAY2RGB)
      #згладжування об'єкта методом Гаусса
      imgThresh = cv2.GaussianBlur(imgThresh, (5, 5), 2)
     #морфологічна операція по розширенню об`єкту для заповнення випадкового простору в
об'єкті
      imgThresh = cv2.dilate(imgThresh, np.ones((5,5),np.uint8))
      #трансформація картинки в сірі відтінки
      imgThresh = cv2.cvtColor(imgThresh, cv2.COLOR_BGR2GRAY)
      #бінарізація зображення
      _, imgThresh = cv2.threshold(imgThresh, 130, 255, cv2.THRESH_BINARY)
      #морфологічні операції для покращення цільності об'єкта на зображені
      st1 = cv2.getStructuringElement(cv2.MORPH  RECT, (21, 21), (10, 10))
      st2 = cv2.getStructuringElement(cv2.MORPH  RECT, (11, 11), (5, 5))
      imgThresh = cv2.morphologyEx(imgThresh, cv2.MORPH  CLOSE, st1)
      imgThresh = cv2.morphologyEx(imgThresh, cv2.MORPH_OPEN, st2)
      #знаходження контурів об'єкта
      img, contours,hierarchy = cv2.findContours(imgThresh, 2, 4)
      #формування прямокутника (зеленого кольору) навколо об'єкта
      cnt = contours[0]
      a,b,w,h = cv2.boundingRect(cnt)
      cv2.rectangle(imgOriginal,(a,b),(a+w,b+h),(0,255,0),2)
      #знаходження центру об'єкта
      x = (a+w)/2
      y = (b+h)/2
      #вивід картинки на екран монітору
      cv2.namedWindow(«imgOriginal», cv2.WINDOW  AUTOSIZE)
      cv2.namedWindow(«imgThresh», cv2.WINDOW_AUTOSIZE)
      cv2.imshow(«imgOriginal», imgOriginal)
      cv2.imshow(«imgThresh», imgThresh)
   # endwhile
   cv2.destroyAllWindows()    #очищення пам'яті від картинки
   return      #знову на початок циклу
#########################################################################################
if   name__ == «__main__»:
   main()
```

Рис. 2.

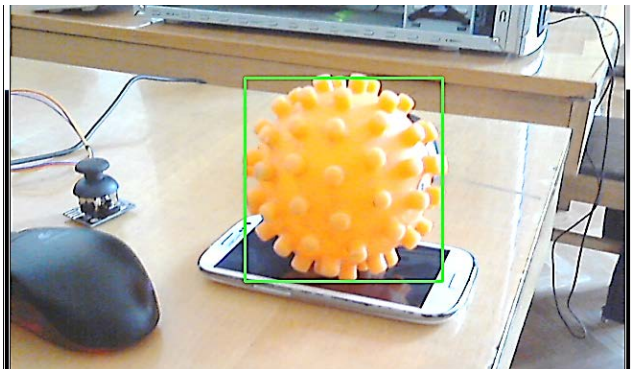

Рис. 3а. Заданий об'єкт

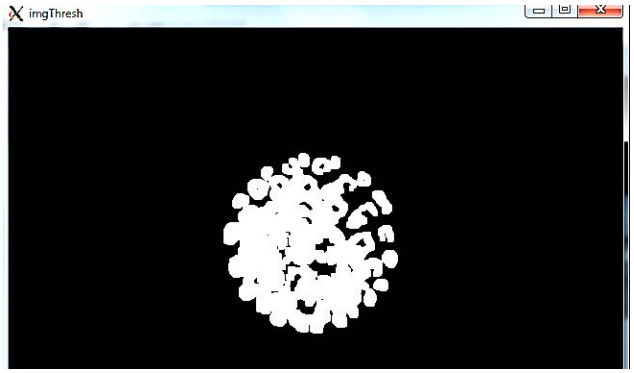

Рис. 3б. Результат розпізнавання об'єкту

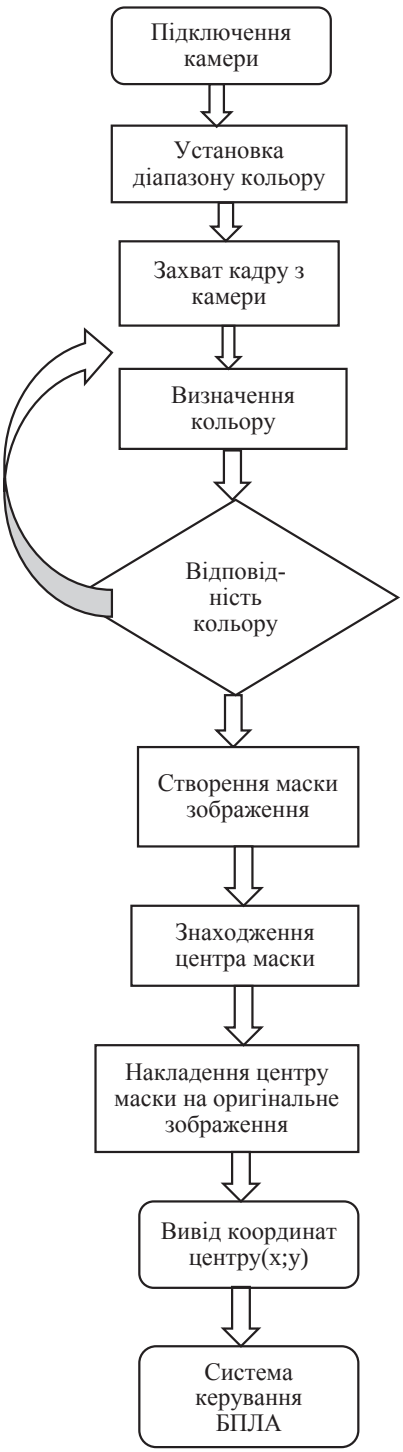

Рис. 4. Блок-схема алгоритму технології розпізнавання зображень за кольором

цього простору є ознаки, за якими здійснюється розпізнавання. Таким чином, координатами об'єктів є значення відповідних ознак. Далі, розпізнавання здійснюється на основі аналізування мір близькості між об'єктами.

Виділяють різні типи ознак: дихотомічні (ознака може бути присутня або відсутня; наприклад, є крила або немає крил); номінальні (наприклад, колір: червоний, синій, зелений і т.п.); порядкові (наприклад, «великий» – «середній» – «маленький»); кількісні. Для кожного типу ознак можна вводити свої міри відстані між об'єктами.

В нашому випадку ми використовуємо у якості координат об'єкту значення номінальних ознак-кольору.

Виділення невирішених раніше частин загальної проблеми. Системи технічного зору на сьогодні набули великого розповсюдження та знаходять все ширше застосування у різноманітних галузях людської діяльності. Важлива особливість сучасного підходу до створення систем технічного зору полягає у швидкому переході від ідеї до математичного алгоритму, перевірка якого не потребує значних затрат. Знадобиться лише ПК та найпростіша веб-камера. Але, не дивлячись, на простоту реалізації, застосування завдяки СТЗ більшість складних технологічних процесів стали автономними, а виконання поставлених завдань-більш швидким та зручним.

Постановка задачі. Ціллю роботи є створення алгоритму та програмного забезпечення, в якому по заданому кольору камера знаходить відповідний об'єкт, розпізнає його, передає сигнали до системи керування та рухається за об'єктом; все проходить в автономному режимі.

У нашій роботі СТЗ використовує технологію ідентифікації символу/знаку/об'єкту за його кольором (при умові, що камера є багатокольоровою).

Викладення основного матеріалу. Для реалізації алгоритму було створено лабораторний стенд, до складу якого входить наступне обладнання:

- мікроконтролер Raspberry Pi 2B;
- мікроконтролер Arduino Mega;
- кроковий двигун з драйвером ULN2003;
- веб-камера Logitech;
- програмні середовища openCV, Python.

Підхід виділення об'єктів на зображеннях з розподілу колірної гами є в своїй основі досить простим, але може використовуватися в ряді випадків. Іноді цей підхід застосовується як первинний етап обробки зображення, для вирішення більш складних завдань, наприклад детектування осіб. Такий

підхід може застосовуватися і в системах трекінгу, якщо колір об'єкта відрізняється від фону.

Рішення завдання розбивається на кілька етапів:

· виділення пікселів, відповідних заданому об'єкту;

· виділення контурами знайдені об'єкти;

· знаходження контуру об'єкту;

· побудова прямокутника, до якого потрапляють усі точки контуру об'єкту.

За допомогою програмного середовища OpenCV та мови програмування Python ми створюємо відповідне програмне забезпечення.

OpenCV – (Open Computer Vision) – бібліотека комп'ютерного зору з відкритим вихідним кодом, що надає набір типів даних і чисельних алгоритмів для обробки зображень алгоритмами комп'ютерного зору. Реалізована на C/C++.

Для виділення та пошуку об'єктів за кольором використовується колірна модель HSV що описує колірний простір, заснований на трьох характеристиках кольору: колірному тоні (Hue), насиченості (Saturation) і яскравості (Brightness, Value). Даний простір кольору є нелінійним. Простір HSV (HSI) можна вважати ідеальним засобом для побудови алгоритмів обробки зображень, оскільки в його основі лежить природний та інтуїтивно зрозумілий людині опис кольору.

Отримати кольоровий тон для моделі HSV можна із простору RGB.

Лістинг коду програми для реалізації алгоритму розпізнавання об'єкту за кольором (см. рис. 2).

На рис. 3 (а, б) зображено об'єкт та результат розпізнавання об'єкту за помаранчевим кольором.

Для представленого алгоритму було використано просторовий метод обробки зображень. Просторова область – це множина пікселів, які складають зображення. Просторові методи – це процедури, що оперують безпосередньо значеннями пікселів зображення. Процеси просторової обробки описуються наступним рівнянням:

$$g(x,y) = T[f(x,y)],$$

де $f(x, y)$ – вхідне зображення, $g(x, y)$ – оброблене зображення, а T – оператор над f, визначений в деякому околі точки (x, y).

Головний підхід у визначенні околу навколо точки (x, y) полягає у використанні квадратної або прямокутної області – підмножини зображення, що центрується в точці (x, y). Центр даної підмножини пересувається від пікселя до пікселя. Оператор T виконується в кожній точці (x, y) даючи в результаті вихідне значення g для даної точки. Процес використовує тільки пікселі усередині області зображення, обмеженої заданим околом.

Збільшення розмірів околу надає значно більшої гнучкості методу. Принцип полягає в тому, що для знаходження значення g в деякій точці (x, y), використовуються значення функції f усередині деякого околу заздалегідь заданої форми, що оточує точку (x, y). Один з основних підходів у такій постановці задачі базується на використанні маски (фільтра) – невеликого двовимірного масиву, значення елементів якого визначають сутність процесу обробки, наприклад, підвищення різкості зображення.

В графічному вигляді алгоритм матиме наступний вигляд (рис. 4).

Висновки. Для системи технічного зору з технологією розпізнавання за кольором були розроблені, реалізовані та протестовані алгоритм та програмне забезпечення.

Створено лабораторний стенд для реалізації та демонстрації алгоритму та програмного забезпечення.

Результати роботи будуть використані на БПЛА, типу квадракоптер, з метою автономного керування пристроєм.

Список літератури:

1. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: Техносфера, 2005. – 1072 с.
2. Хорн Б. К.П. Зрение роботов. – М.: Мир. – 1989. – 400 с.
3. Hoffmann G. CIE Color Space. Режим доступу: http://www.fho-emden.de/~hoffmann/ciexyz29082000.pdf. – Перевірено 05.09.2011.
4. Шапиро Л., Стокман Дж. Компьютерное зрение / Пер. с англ. – М.: БИНОМ. Лаборатория знаний. – 2006. – 752 с.
5. Дзюба В. Г., Варфоломєєв А. Ю. Системи технічного зору / Навчальне видання. – К.: НТУУ «КПІ», 2011 р. – 148 с.
6. http://studopedia.org/6-105311.html

Котвицкий Р.С., Сарыбога А.В.
Национальный технический университет Украины
«Киевский политехнический институт»

СИСТЕМА ТЕХНИЧЕСКОГО ЗРЕНИЯ В ЗАДАЧАХ СТАБИЛИЗАЦИИ БЕСПИЛОТНОГО ЛЕТАТЕЛЬНОГО АППАРАТА

Аннотация
Работа посвящена разработке системы технического зрения с целью стабилизации и управления беспилотным летательным аппаратом. Приведен алгоритм обработки изображения. Разработана общая архитектура программного обеспечения для реализации системы, а также описана типичная задача для технического зрения.

Ключевые слова: микроконтроллер, система технического зрения, распознавание образов.

Kotvitsky R.S., Saryboga G.V.
National Technical University of Ukraine
«Kiev Polytechnic Institute»

SYSTEM OF TECHNICAL VISION IN PROBLEMS OF STABILIZATION OF A PILOTLESS VEHICLE

Summary
This work is dedicated to questions of development of a machine vision system for the needs of navigation and control of a pilotless vehicle. Algorithms of objects recognition are presented. General architecture of software needed for realization of the system is supplied. Typical challenge for machine vision are described.
Keywords: microcontroller, system of technical vision, image recognition.