Министерство образования и науки Российской федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Алтайский государственный технический университет им. И. И.Ползунова»

Факультет информационных технологий

Кафедра прикладной математики

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

«\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2015 г.

Отчет по лабораторной работе №7

Частичная инвариантность дескрипторов к аффинным трансформациям «Интеллектуальные технологии обработки изображений»

ИЗ 231000.3.000 О

Студент группы 8ПИ-41 А.Ю. Смирнова

(И.О. Фамилия)

Преподаватель старший преподаватель М.Г. Казаков

(должность, ученое звание) (И.О. Фамилия)

Барнаул 2015

**Цель работы**

Познакомиться с возможностью достижения относительной инвариантности дескрипторов к аффинным трансформациям изображения.

**Формулировка задачи**

* Реализовать распределение значений градиентов по смежным гистограммам, добавить весовые коэффициенты исходя из расстояния до соответствующих центров.
* Опционально – локальное уточнение центра дескриптора
* Оценить работу дескриптора исходя из искажений поворота плоскости, оценить повторяемость в этом случае.
* Сравнить полученные результаты, сравнить с полученными в шестой работе.

**Исходый код:**

**Результаты работы программы:**

descriptorgenerator.cpp

shared\_ptr<Descriptor> DescriptorGenerator::getDescriptor(Point p, int surSize, int gistNum, int beansNum)

{

//find point angle

auto descriptor = make\_shared<Descriptor>(beansNum\*gistNum, p);

surSize = surSize\*p.innerScale;

int gistSize = ceil(surSize/gistNum);

int curGistNum;

float weight,angle,r;

float oneBean = 360 / beansNum;

int first, second;

float firstValue, secondValue;

float sigma = surSize\*0.5;

float dx, dy;

float radA = p.angle \* M\_PI / 180;

int curGistX, curGistY;

float topLeftCenterX, topLeftCenterY;

for(int i = -surSize/2 - sqrt(2); i < surSize/2 + sqrt(2); i++)

for(int j = -surSize/2 - sqrt(2); j < surSize/2 + sqrt(2); j++)

{

weight = magnitudes->getPixel(p.y+i, p.x+j)\*(pow(M\_E,-(i\*i + j\*j)/(2\*sigma\*sigma)))/(2\*M\_PI\*sigma\*sigma);

angle = - p.angle + angles->getPixel(p.y+i, p.x+j);

while(angle < 0) angle += 360;

dx = j \* cos(radA) + i\*sin(radA);

dy = - j \* sin(radA) + i\*cos(radA);

curGistX = (int)(dy + surSize/2) / gistSize;

curGistY = (int)(dx + surSize/2) / gistSize;

if(dx < curGistX\*gistSize + gistSize/2) curGistX = (curGistX - 1 + gistNum)%gistNum;

if(dy > curGistY\*gistSize + gistSize/2) curGistY = (curGistY - 1 + gistNum)%gistNum;

topLeftCenterX = curGistX\*gistSize + gistSize/2;

topLeftCenterY = curGistY\*gistSize + gistSize/2;

//add in top left

curGistNum = (curGistY)\*gistNum + curGistX;

descriptor->addInGist(curGistNum, angle, weight\*(gistSize - dy + topLeftCenterY)\*(gistSize - dx + topLeftCenterX) / pow(gistSize, 2), beansNum);

//add in top right

curGistNum = (curGistY)\*gistNum + (curGistX + 1)%gistNum;

descriptor->addInGist(curGistNum, angle, weight\*(gistSize - dy + topLeftCenterY)\*(dx - topLeftCenterX) / pow(gistSize, 2), beansNum);

//add in bottom left

curGistNum = ((curGistY + 1) % gistNum)\*gistNum + curGistX;

descriptor->addInGist(curGistNum, angle, weight\*(dy - topLeftCenterY)\*(gistSize - dx + topLeftCenterX) / pow(gistSize, 2), beansNum);

//add in bottom right

curGistNum = ((curGistY + 1) % gistNum)\*gistNum + (curGistX + 1)%gistNum;

descriptor->addInGist(curGistNum, angle, weight\*(dy - topLeftCenterY)\*(dx - topLeftCenterX) / pow(gistSize, 2), beansNum);

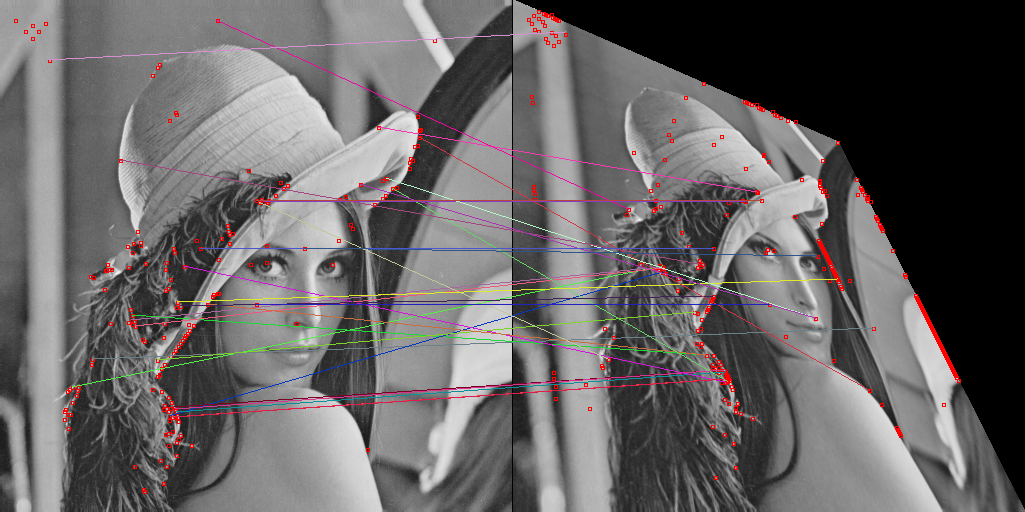
}

descriptor->normalize();

return descriptor;

}

**Результаты работы программы:**

****