Vision – IF6083

Tugas Homework 4 (HW4)

Reza Budiawan
33221040

Program Studi S3 Teknik Elektro & Informatika

INSTITUT TEKNOLOGI BANDUNG
Desember 2022

Indeks Konten

Deskripsi Umum Tugas 4

Pada HW4, dilakukan operasi optical flow terhadap citra yang diolah.

4.1: Integral images

Deskripsi

Tuliskan body method dari “make_integral_image” dengan membuat proses summed-area table. Proses ini akan menghasilkan jumlah dari rectangular subset dari area yang ditentukan.

Solusi

```
image make_integral_image(image im)
{
    image integ = copy_image(im);

    for (int i = 0; i < im.c; ++i){
      for (int j = 1; j < im.h; ++j){
        integ.data[i*im.h*im.w+j*im.w] += integ.data[i*im.h*im.w+(j-1)*im.w];
      }
      for (int k = 1; k < im.w; ++k){
        integ.data[i*im.h*im.w+k] += integ.data[i*im.h*im.w+k-1];
      }
    }
    for (int i = 0; i < im.c; ++i){
      for (int j = 1; j < im.h; ++j){
        for (int k = 1; k < im.w; ++k){
          int idx = i*im.h*im.w + j*im.w + k;
          integ.data[idx] += integ.data[idx-1] + integ.data[idx-im.w] - integ.data[idx-
im.w-1];
        }
      }
    }
    return integ;
}
```

4.2: Smoothing using integral images

Deskripsi

Implementasikan method “box_filter_image” sehingga setiap piksel pada output merupakan rataan dari piksel pada window size yang diberikan (s).

Solusi

```
image box_filter_image(image im, int s)
{
    int i,j,k;
    int offset = s / 2;
```

```
    image integ = make_integral_image(im);
    image S = make_image(im.w, im.h, im.c);
    // TODO: fill in S using the integral image.
    for(i = 0; i < im.w; i++) {
        for(j = 0; j < im.h; j++) {
            for(k = 0; k < im.c; k++) {
                float sum = get_pixel(integ, i + offset, j + offset, k) -
                            get_pixel(integ, i + offset, j - offset - 1, k) -
                            get_pixel(integ, i - offset - 1, j + offset, k) +
                            get_pixel(integ, i - offset - 1, j - offset - 1, k);
                set_pixel(S, i, j, k, sum / (s * s));
            }
        }
    }
    return S;
}
```

4.3: Lucas-Kanade optical flow

Pada task ini, akan diimplementasikan Lucas-Kanade optical flow. Perhitungan yang dilakukan adalah sebagai berikut:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i)I_y(q_i) \\ \sum_i I_y(q_i)I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i)I_t(q_i) \\ -\sum_i I_y(q_i)I_t(q_i) \end{bmatrix}$$

4.3.1: Time-structure matrix

Deskripsi

Untuk flow equation diperlukan informasi spatial & temporal gradient. Untuk itu diperlukan perhitungan time-structure matrix. Hal ini dituliskan pada method “time_structure_matrix”.

Solusi

```
image time_structure_matrix(image im, image prev, int s)
{
    int i;
    int converted = 0;
    if(im.c == 3){
        converted = 1;
        im = rgb_to_grayscale(im);
        prev = rgb_to_grayscale(prev);
    }

    // TODO: calculate gradients, structure components, and smooth them
    image S = make_image(im.w, im.h, 5);
    image gx_filter = make_gx_filter();
```

```
    image gy_filter = make_gy_filter();
    image gx = convolve_image(im, gx_filter, 0);
    image gy = convolve_image(im, gy_filter, 0);
    image gt = make_image(im.w, im.h, 1);
    for (i = 0; i < im.h*im.w; ++i){
      gt.data[i] = im.data[i] - prev.data[i];
    }
    for (i = 0; i < im.h*im.w; ++i){
      S.data[i] = gx.data[i] * gx.data[i];
      S.data[i+im.h*im.w] = gy.data[i] * gy.data[i];
      S.data[i+2*im.h*im.w] = gx.data[i] * gy.data[i];
      S.data[i+3*im.h*im.w] = gx.data[i] * gt.data[i];
      S.data[i+4*im.h*im.w] = gy.data[i] * gt.data[i];
    }
    S = box_filter_image(S, s);

    if(converted){
        free_image(im); free_image(prev);
    }
    free_image(gx_filter);
    free_image(gy_filter);
    free_image(gx);
    free_image(gy);
    free_image(gt);
    return S;
}
```

4.3.2: Calculating velocity from the structure matrix

Deskripsi

Tuliskan body method “velocity_image” untuk menghitung velocity tiap piksel pada x & y direction. Gunakan matrix M (dan lakukan inversi) untuk melakukan perhitungan.

Solusi

```
image velocity_image(image S, int stride)
{
    image v = make_image(S.w/stride, S.h/stride, 3);
    int i, j;
    matrix M = make_matrix(2,2);
    for(j = (stride-1)/2; j < S.h; j += stride){
        for(i = (stride-1)/2; i < S.w; i += stride){
            float Ixx = S.data[i + S.w*j + 0*S.w*S.h];
            float Iyy = S.data[i + S.w*j + 1*S.w*S.h];
            float Ixy = S.data[i + S.w*j + 2*S.w*S.h];
            float Ixt = S.data[i + S.w*j + 3*S.w*S.h];
            float Iyt = S.data[i + S.w*j + 4*S.w*S.h];
```

```
            // TODO: calculate vx and vy using the flow equation
             matrix T = make_matrix(2,1);
            T.data[0][0] = -Ixt;
            T.data[1][0] = -Iyt;
            double arr[2][2] = {{Ixx, Ixy}, {Ixy, Iyy}};
            memcpy(*M.data, arr[0], sizeof(arr[0]));
            memcpy(*(M.data+1), arr[1], sizeof(arr[1]));

            matrix Minv = matrix_invert(M);
            if(!Minv.data) continue;
            matrix V = matrix_mult_matrix(Minv, T);
            float vx = V.data[0][0];
            float vy = V.data[1][0];
            set_pixel(v, i/stride, j/stride, 0, vx);
            set_pixel(v, i/stride, j/stride, 1, vy);
        }
    }
    free_matrix(M);
    return v;
}
```

Hasil

Jalankan script berikut (yang juga tercantum pada tryhw4.py):

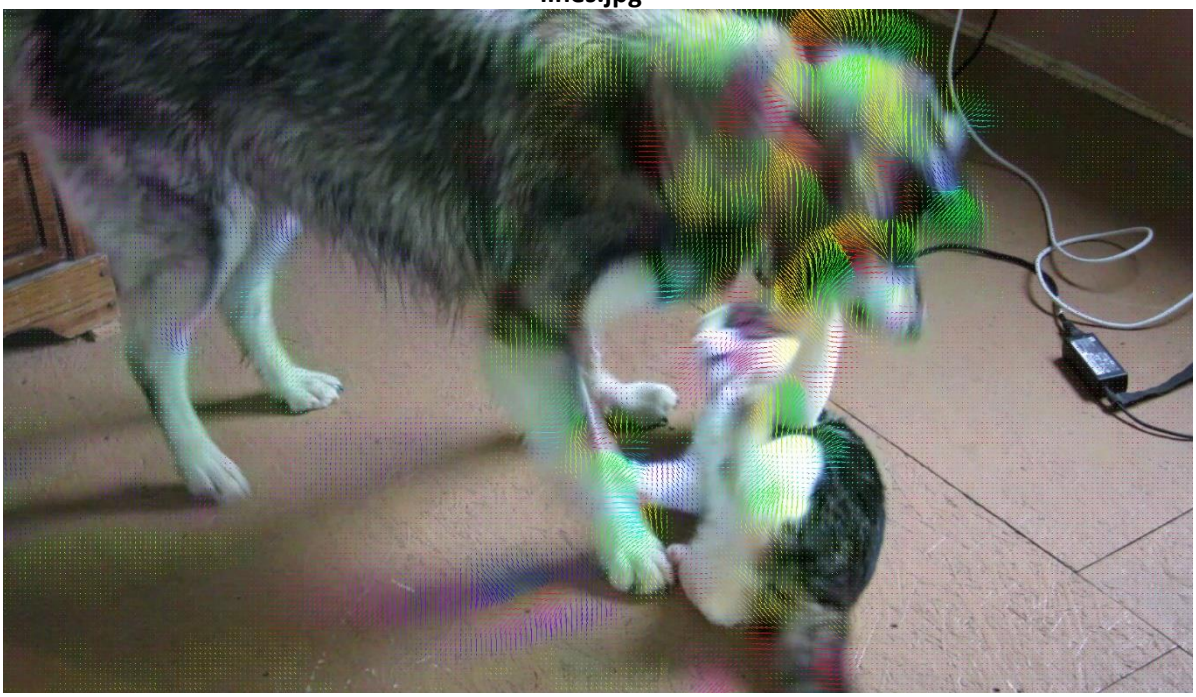
```
from uwimg import *

a = load_image("data/dog_a.jpg")
b = load_image("data/dog_b.jpg")
flow = optical_flow_images(b, a, 15, 8)
draw_flow(a, flow, 8)
save_image(a, "lines")
```

Script di atas menghitung optical flow antara 2 gambar (dog_a.jpg & dog_b.jpg) dengan citra hasil pengolahan adalah lines.jpg.

dog_a.jpg

dog_b.jpg

lines.jpg



4.4: Optical flow demo using OpenCV

Deskripsi

Install OpenCV untuk mendapatkan citra dari webcam & tampilkan hasilnya secara realtime. Terdapat 3 hal yang perlu dilakukan: install OpenCV, enable OpenCV compilation pada makefile, dan run “tryhw4.py”.

Solusi

Hal ini tidak dapat dilakukan karena error berikut pada makefile:

```
(computervision) C:\Users\rezab\Documents\Studi Lanjut\Sem 3\Vision - IF6083\If6083ComputerVision>make
g++ -std=c++11  -Iinclude/ -Isrc/  -DOPENCV `pkg-config --cflags opencv4` -Wall -Wno-unknown-pragmas -Wfatal-errors -fPI
C  -Ofast -DOPENCV -c ./src/image_opencv.cpp -o obj/image_opencv.o
g++: error: unrecognized command-line option '--cflags'
make: *** [obj/image_opencv.o] Error 1
```