Mata Kuliah : Computer Vision
NIM : L200180136
Nama : Muh. Riza Radyaka S.

Tanggal Ujian : Sabtu, 30 Januari 2021
Dosen Pengampu : Bana Handaga, Ir. M.T., Ph.D

Ujian Akhir Semester Computer Vision

1. Pengertian Fitur Gambar

   Fitur yang berarti bidang minat dalam gambar yang unik dan mudah dikenali. Salah satu fitur yang ada digambar adalah tepi dan blob (wilayah gambar yang sangat berbeda dari area sekitar).

   Algoritma fitur gambar : Harris, SIFT, SURF, FAST, BRIEF, ORB

   Harris

   Kelebihan : bisa mendeteksi sudut karena sudut adalah sudut

   Kekurangan : jika gambar dalam skala kecil atau besar, beberapa bagian gambar kehilangan kualitas sudut

   SIFT

   Kelebihan : bisa mendeteksi gambat terlepas dari skala gambar

   Kekurangan : lebih lambat dibanding SURF

   SURF

   Kelebihan : mendeteksi dengan keypoint Fast Hessian, lebih cepat daripada SIFT

   Kekurangan : SURF hanya sebuah keypoint

   FAST

   Kelebihan : alternatif lebih cepat dari SIFT dan SURF

   Kekurangan : hanya sebagai keypoint

   BRIEF

   Kelebihan : deskriptor tercepat dengan serangkaian pengoptimalan yang sangat baik untuk pencocokan fitur

   Kekurangan : hanya sebagai deskriptor gambar,

   ORB

   Kelebihan : merupakan gabungan dari keypoint FAST dan deskriptor BRIEF

   Kekurangan : harus menggunakan konsep keypoint dan deskriptor

2. Objek gambar orang berjalan
   Langkah - langkah membangun aplikasi deteksi orang berjalan:
   - Melakukan implementasi pembuatan persegi panjang dalam berada didalam objek persegi panjang lainnya. Dengan membuat fungsi is_inside(a,b). a nilai persegi panjang dalam, dan b nilai persegi panjang luar. Jika nilai a didalam b makan mengembalikan True, jika tidak mengembalikan nilai False
     Kode program:

```
import cv2

def is_inside(a, b):

        ax, ay, aw, ah = a

        bx, by, bw, bh = b

        return ax > bx and ax + aw < bx + bw and ay > by and ay + ah < by + bh
```

   - Kemudian membuat instance cv2.HOGDescriptor dan memanggilkan setSVMDetector untuk mendapatkan label SVM detektor orang. Kemudian, melakukan memasukan gambar yang akan di seleksi dalam kasus ini orang berjalan. Hasil dari HOGDescriptor akan menghasilkan dua buah output, yaitu persegi panjang bersarang dan bobot keyakinan objek terdeteksi
     Kode program :

```
hog = cv2.HOGDescriptor()

hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())



img = cv2.imread('../images/orang_berjalan.jpg')

found_rects, found_weights = hog.detectMultiScale(

   img, winStride=(4, 4), scale=1.02, finalThreshold=1.9)
```

   - Setelah itu, dapat difilter hasil deteksi untuk menghapus persegi panjang yang bersarang. Dengan membuat fungsi loop is_inside bersarang
     Kode program :

```
found_rects_filtered = []

found_weights_filtered = []

for ri, r in enumerate(found_rects):

        for qi, q in enumerate(found_rects):

                if ri != qi and is_inside(r, q):
```

```
            break
        else:
            found_rects_filtered.append(r)
            found_weights_filtered.append(found_weights[ri])
```

- Terkakhir, tinggal menampilkan hasil visualisasi dari gambar persegi panjang dan bobot untuk mendeteksi objek orang :
  Kode program :

```
for ri, r in enumerate(found_rects_filtered):
    x, y, w, h = r
    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 255), 2)
    text = '%.2f' % found_weights_filtered[ri]
    cv2.putText(img, text, (x, y - 20), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255, 255), 2)


cv2.imshow('Orang Berjalan Detected', img)
cv2.imwrite('./orang_bejalan_detected.jpg', img)
cv2.waitKey(0)
```

3. Konsep sistem tracking object adalah menangkap bingkai atau frame pada sebuah objek yang dipilih secara berurutan atau terus menerus.
   Langkah – langkah pembuatan Object Tracking Pejalan Kaki :
   - Merencanakan alur aplikasi dengan menangkap bingkai/frame, gunakan 20 frame pertama untuk pengurangan latar belakang, frame selanjutnya gunakan sebagai identifikasi object latar depan yang akan bergerak, terakhir lacak setiap pejalan kaki dengan filter Kalman dan MeanShift di frame seterusnya.
   - Menerapkan kelas pedestrian dan menerapkan filter Kalman yang dapat meprediksi posisi suatu objek berdasarkan pengamatan historis dan dapat mengoreksi prediksi berdasarkan data sebenarnya, tetapi filter ini hanya dapat mendapatkan satu buah objek
     Kode program :

```
import cv2
import numpy as np
class Pedestrian():
  """A tracked pedestrian with a state including an ID, tracking
  window, histogram, and Kalman filter.
```

```
"""
def __init__(self, id, hsv_frame, track_window):
        self.id = id
        self.track_window = track_window
        self.term_crit = \
           (cv2.TERM_CRITERIA_COUNT | cv2.TERM_CRITERIA_EPS, 10, 1)
        # Initialize the histogram.
        x, y, w, h = track_window
        roi = hsv_frame[y:y+h, x:x+w]
        roi_hist = cv2.calcHist([roi], [0], None, [16], [0, 180])
        self.roi_hist = cv2.normalize(roi_hist, roi_hist, 0, 255, cv2.NORM_MINMAX)
        # Initialize the Kalman filter.
        self.kalman = cv2.KalmanFilter(4, 2)
        self.kalman.measurementMatrix = np.array(
          [[1, 0, 0, 0],
          [0, 1, 0, 0]], np.float32)
        self.kalman.transitionMatrix = np.array(
          [[1, 0, 1, 0],
          [0, 1, 0, 1],
          [0, 0, 1, 0],
          [0, 0, 0, 1]], np.float32)
        self.kalman.processNoiseCov = np.array(
          [[1, 0, 0, 0],
          [0, 1, 0, 0],
          [0, 0, 1, 0],
          [0, 0, 0, 1]], np.float32) * 0.03
        cx = x+w/2
        cy = y+h/2
```

```
self.kalman.statePre = np.array(
  [[cx], [cy], [0], [0]], np.float32)
self.kalman.statePost = np.array(
  [[cx], [cy], [0], [0]], np.float32)
def update(self, frame, hsv_frame):
  back_proj = cv2.calcBackProject([hsv_frame], [0], self.roi_hist, [0, 180], 1)
  ret, self.track_window = cv2.meanShift(
     back_proj, self.track_window, self.term_crit)
  x, y, w, h = self.track_window
  center = np.array([x+w/2, y+h/2], np.float32)
prediction = self.kalman.predict()
estimate = self.kalman.correct(center)
center_offset = estimate[:,0][:2] - center
self.track_window = (x + int(center_offset[0]),y + int(center_offset[1]), w, h)
x, y, w, h = self.track_window
# Draw the predicted center position as a circle.
cv2.circle(frame, (int(prediction[0]), int(prediction[1])), 4, (255, 0, 0), -1)
# Draw the corrected tracking window as a rectangle.
cv2.rectangle(frame, (x,y), (x+w, y+h), (255, 255, 0), 2)
# Draw the ID above the rectangle.
cv2.putText(frame, 'ID: %d' % self.id, (x, y-5),
   cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 0, 0),1, cv2.LINE_AA)
```

- Menerapkan fungsi utama dari memuat file video, kernel morfologi, data pedestrian, iterasi loop pengurangan latar belakang, membuat thresholded, koversi frame ke format HSV dalam MeanShift, membuat bingkai dan daftar pedestrian update, menampilkan hasil pelacakan
  Kode program :

```
def main():
cap = cv2.VideoCapture('pedestrians.avi')
# Create the KNN background subtractor.
```

```
bg_subtractor = cv2.createBackgroundSubtractorKNN()
history_length = 20
bg_subtractor.setHistory(history_length)
erode_kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (3, 3))
dilate_kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (8, 3))
pedestrians = []
num_history_frames_populated = 0
while True:
        grabbed, frame = cap.read()
        if (grabbed is False):
                break
# Apply the KNN background subtractor.
fg_mask = bg_subtractor.apply(frame)
# Let the background subtractor build up a history.
if num_history_frames_populated < history_length:
        num_history_frames_populated += 1
        continue
# Create the thresholded image.
 _, thresh = cv2.threshold(fg_mask, 127, 255, cv2.THRESH_BINARY)
cv2.erode(thresh, erode_kernel, thresh, iterations=2)
cv2.dilate(thresh, dilate_kernel, thresh, iterations=2)
# Detect contours in the thresholded image.
contours, hier = cv2.findContours(thresh, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
hsv_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV) # Draw rectangles around
large contours.
# Also, if no pedestrians are being tracked yet, create some.
should_initialize_pedestrians = len(pedestrians) == 0
id = 0
```

```
for c in contours:
        if cv2.contourArea(c) > 500:
            (x, y, w, h) = cv2.boundingRect(c)
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 1)
                if should_initialize_pedestrians:
                    pedestrians.append( Pedestrian(id, frame, hsv_frame, (x, y,
                w, h)))
         id += 1
# Update the tracking of each pedestrian.
     for pedestrian in pedestrians:
      pedestrian.update(frame, hsv_frame)
cv2.imshow('Pedestrians Tracked', frame)
      k = cv2.waitKey(110)
      if k == 27: # Escape
        break


if __name__ == "__main__":
  main()
```

4. Pengertian ANN

ANN atau Artificial Neural Network adalah sebuah teknik atau cara pendekatan pengolahan informasi yang terinspirasi oleh cara kerja dari sistem saraf manusia (biologis). Bedanya ANN dengan DNN adalah DNN merupakan lapisan-lapisan dari ANN. Sehingga DNN merupakan bagian dari ANN.

Resume Deep Learning

Pada link tersebut membahas tentang Deep Learning With OpenCV. Dimulai dengan membuat klasifikasi tentang gambar yang akan digunakan untuk deep learning. Kemudian mendapatkan label kelas yang akan disimpan dalam row dan dipisah dengan koma. Dan membuat label itu menjadi normal sebagai blob image. Terus diubah blob tersebut menjadi input. Dan menampilkannya di tempat sesuai dengan labelnya. Dan akhirnya masukan gambar tersebut untuk mendapatkan hasil dengan probabilitas serta label kelasnya.

5. Deep Neural Network

   Dari tiga link tersebut merupakan open source framework, open source platform, dan open source aplikasi. Tiga hal tersebut adalah PyTorch, TensorFlow, dan Anaconda. Pada pembelajaran Deep Neural Network menggunnakan TensorFlow dengan package bernama Keras. Karena pada TensorFlow sendiri sebenarnya sudah cukup, tetapi lebih mudah jika kita menggunakan package Keras. TensorFlow juga memiliki fungsionalitas yang cukup luas untuk machine learning lebih tepatnya Deep Neural Network. Sedangkan PyTorch, adalah framework yang bisa membantu menambahkan fungsionalitas pada Deep Neural Network yang akan dibuat. Tetapi Pytorch memerlukan Anaconda sebagai aplikasi yang bisa menjalankan platform serta framework tersebut.