Exercise8

November 9, 2021

1 CS-E4850 Computer Vision Exercise Round 8

The problems should be solved before the exercise session and solutions returned via MyCourses. For this exercise round you should return a pdf file containing written answers to the questions below.

1.0.1 Exercise 1. Face tracking example using KLT tracker

Run the example as instructed below and answer the questions.

a) Run `Exercise8.ipynb`
b) Run `Exercise8.ipynb` with a different input by changing the input to obama.avi: `frames=faceTracker('obama.avi')`
c) What could be the main reasons why most of the features are not tracked very long in case b) above?
d) How could one try to avoid the problem of gradually losing the features? Suggest one or more improvements.
e) Voluntary task: Capture a video of your own face or of a picture of a face, and check that whether the tracking works for you. That is, replace the input video path in `faceTrackingDemo.py` with the path to your own video.

1.0.2 Exercise 2. Kanade-Lucas-Tomasi (KLT) feature tracking (Pen & paper problem)

Read Sections 2.1 and 2.2 from the paper by Baker and Matthews. Show that the Equation (10) in the paper gives the same solution as the equations on slide 25 of Lecture 7, when the geometric warping W (between the current frame and the template window in the previous frame) is a translation.

1.1 Anwser Sheet:

1.1.1 c) What could be the main reasons why most of the features are not tracked very long in case b) above?

Keypoints of corners will be missed because the image is rotated or the tracking object is moving too fast. KLT algorithm requires minimal movement(The movement of pixels on the image changes slowly with time.) and spatial consistency(adjacent points on the same surface in the scene have similar motion and projections onto the image plane are relatively close together.)

1.1.2 d) How could one try to avoid the problem of gradually losing the features? Suggest one or more improvements.

We can try to avoid large movement in a short period of time, letting algorithm dececting and tracking new features. We can also keep outliers features for longer time. But this leads to lower performance.

[1]:
```
# This cell is used for creating a button that hides/unhides code cells to␣
 ↪quickly look only the results.
# Works only with Jupyter Notebooks.

import os
from IPython.display import HTML

HTML('''<script>
code_show=true;
function code_toggle() {
if (code_show){
$('div.input').hide();
} else {
$('div.input').show();
}
code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
<form action="javascript:code_toggle()"><input type="submit" value="Click here␣
 ↪to toggle on/off the raw code."></form>''')
```

[1]:
```
<IPython.core.display.HTML object>
```

[1]:
```
# Description:
#   Exercise8 python demo.
#
# Copyright (C) 2018 Santiago Cortes, Juha Ylioinas, Tapio Honka
#
# This software is distributed under the GNU General Public
# Licence (version 2 or later); please refer to the file
# Licence.txt, included with the software, for details.

import matplotlib.pyplot as plt
import matplotlib.animation as animation
from IPython.display import HTML
from faceTrackingDemo import faceTracker
```

```
The data directory is /coursedata
Data stored in /coursedata/exercise-08-data
```

[2]:
```
%%capture
fig = plt.figure(figsize=(10,10))

# frames of the processed input video
# change the input to obama.avi in part b)
# frames = faceTracker('santi.avi')

frames = faceTracker('obama.avi')

# create an animation that can be embedded in the notebook
ani = animation.ArtistAnimation(fig, frames, interval=50, blit=True,␣
 ↪repeat_delay=2000)
```

[3]:
```
display(HTML(ani.to_html5_video()))
```

```
<IPython.core.display.HTML object>
```

[ ]:
```

```

Yan Gengcong
1009903

2:

Equation in (10):

$$\Delta p = H^{-1} \sum_x \left[\nabla I \frac{\partial W}{\partial p} \right]^T \left[T(x) - I(W(x;p)) \right]$$

And where $\Delta p = \begin{bmatrix} u \\ v \end{bmatrix}$, $H = \sum_x \left[\nabla I \frac{\partial W}{\partial p} \right]^T \left[\nabla I \frac{\partial W}{\partial p} \right]$

$\nabla I = \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right]$

We know that:

$$\frac{\partial W}{\partial p} = \begin{bmatrix} \frac{\partial W_x}{\partial p_1} & \frac{\partial W_x}{\partial p_2} \\ \frac{\partial W_y}{\partial p_2} & \frac{\partial W_y}{\partial p_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\therefore H = \sum_x \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix} \cdot \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix}$$

So equation in (10) can be written:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix}^{-1} \cdot \sum_x \begin{bmatrix} I_x \\ I_y \end{bmatrix} \left[T(x) - I(W(x;p)) \right]$$

$$\Rightarrow \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_x I_x I_t \\ \sum_x I_y I_t \end{bmatrix}$$

which is the same equation on slide 25 of lecture 7.