RBE549 - Final Exam

Keith Chester

Due date: December 12, 2022

Problem 1

In this problem, we are tasked with solving an iterative optical flow problem. To compute optical flow, we learned an interative method to update $u(x, y)$, $v(x, y)$ at each iteration, according to:

$$\begin{bmatrix} u(x,y) \\ v(x,y) \end{bmatrix}^2 = \begin{bmatrix} \lambda I_x^2 + 4 & \lambda I_x I_y \\ \lambda I_x I_y & \lambda I_y^2 + 4 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{n \epsilon neighbors(x,y)} u^{old}(n) - \lambda I_x I_t \\ \sum_{n \epsilon neighbors(x,y)} v^{old}(n) - \lambda I_y I_t \end{bmatrix} \tag{1}$$

We are asked to consider a local coordinate frame (x', y') where x' is aligned with the image gradient and y' is perpendicular to the image gradient. Likewise, $(u', v') = (\frac{dx'}{dt}, \frac{dy'}{dt})$ are the image velocities in this frame. In this coordinate frame,

$$I_{x'} = \sqrt{I_x^2 + I_y^2} \ and \ I_{y'} = 0 \tag{2}$$

We wish to show that the update equations:

$$\begin{bmatrix} u'(x,y) \\ v'(x,y) \end{bmatrix}^2 = \begin{bmatrix} \lambda I_{x'}^2 + 4 & \lambda I_{x'} I_{y'} \\ \lambda I_{x'} I_{y'} & \lambda I_{y'}^2 + 4 \end{bmatrix}^{-1} \begin{bmatrix} \sum_{n \epsilon neighbors(x,y)} u'^{old}(n) - \lambda I_{x'} I_t \\ \sum_{n \epsilon neighbors(x,y)} v'^{old}(n) - \lambda I_{y'} I_t \end{bmatrix} \tag{3}$$

...can reduce to

$$u'^{new}(x,y) = \bar{u}'^{old} - \frac{I_{x'}^2 \bar{u}'^{old} + I_{x'} I_t}{I_{x'}^2 + \frac{4}{\lambda}} \tag{4}$$

$$v'^{new}(x,y) = \bar{v}'^{old} \tag{5}$$

. To do this, we first expand our second term to an equivalnet form to make matters easier for us to work with. Specifically, the inverse of a $2x2$ matrix is:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} \tag{6}$$

...and thus...

$$\begin{bmatrix} \lambda I_{x'}^2 + 4 & \lambda I_{x'} I_{y'} \\ \lambda I_{x'} I_{y'} & \lambda I_{y'}^2 + 4 \end{bmatrix}^{-1} = \frac{1}{(\lambda I_{x'}^2 + 4)(\lambda I_{y'}^2 + 4) - \lambda I_{x'} I_{y'} \lambda I_{x'} I_{y'}} \begin{bmatrix} \lambda I_{y'}^2 + 4 & -\lambda I_{x'} I_{y'} \\ -\lambda I_{x'} I_{y'} & \lambda I_{y'}^2 + 4 \end{bmatrix} \tag{7}$$

...which simplifies to:

$$\frac{1}{4\lambda I_{x'}^2 + 4\lambda I_{y'}^2 + 16} \begin{bmatrix} \lambda I_{y'}^2 + 4 & -\lambda I_{x'} I_{y'} \\ -\lambda I_{x'} I_{y'} & \lambda I_{x'}^2 + 4 \end{bmatrix} \tag{8}$$

This in turn leads us to an expanded view from earlier:

$$u'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \begin{bmatrix} \lambda I_{y'}^2 + 4 & -\lambda I_{x'} I_{y'} \end{bmatrix} \begin{bmatrix} 4\bar{u}'^{old} - \lambda I_{x'} I_t \\ 4\bar{v}'^{old} - \lambda I_{y'} I_t \end{bmatrix} \tag{9}$$

...We can then expand it into this:

$$u'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)}\big((\lambda I_{y'}^2 + 4)(4\bar{u}'^{old} - \lambda I_{x'} I_t) - (\lambda I_{x'} I_{y'})(4\bar{v}'^{old} + \lambda I_{x'} I_{y'} \lambda I_{y'} I_t)\big) \tag{10}$$

$$u'^{new}(x,y) = \frac{\lambda I_{y'}^2 4\bar{u}^{old} - \lambda I_{y'}^2 \lambda I_{x'} I_t + 16\bar{u}'^{old} - 4\lambda I_{x'} I_t - (\lambda I_{x'} I_{y'})4\bar{v}'^{old} + \lambda^2 I_{y'}^2 I_{x'} I_t}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{11}$$

$$u'^{new}(x,y) = \frac{4(\lambda I_{y'}^2 \bar{u}'^{old} + 4\bar{u}'^{old} - \lambda I_{x'} I_t - \lambda I_{x'} I_{\prime} y \bar{v}'^{old})}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{12}$$

$$u'^{new}(x,y) = \frac{(\lambda I_{y'}^2 \bar{u}'^{old} + 4\bar{u}'^{old} - \lambda I_{x'} I_t - \lambda I_{x'} I_{y'} \bar{v}'^{old})}{(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{13}$$

Now we can introduce terms to try and help us simplify to our end goal. To this end, we will be adding in the terms $\lambda I_{x'}^2 \bar{u}'^{old} - \lambda I_{x'}^2 \bar{u}'^{old}$. Since the terms cancel out on their own it's the equivalent of adding 0, and thus does not change our definition.

$$u'^{new}(x,y) = \frac{\lambda I_{y'}^2 \bar{u}'^{old} + 4\bar{u}'^{old} - \lambda I_{x'} I_t - \lambda I_{x'} I_{y'} \bar{v}'^{old} + \lambda I_{x'}^2 \bar{u}'^{old} - \lambda I_{x'}^2 \bar{u}'^{old}}{(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{14}$$

$$u'^{new}(x,y) = \frac{\bar{u}'^{old}\big(\lambda I_{y'}^2 + 4 + \lambda I_{x'}^2\big) - \lambda I_{x'}^2 \bar{u}'^{old} - \lambda I_{x'} I_{y'} \bar{v}'^{old} - \lambda I_{x'} I_t}{(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{15}$$

$$u'^{new}(x,y) = \bar{u}'^{old} - \frac{\lambda I_{x'}^2 \bar{u}'^{old} + \lambda I_{x'} I_{y'} \bar{v}'^{old} + \lambda I_{x'} I_t}{(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \tag{16}$$

As specified before, $I_{y'} = 0$, so we can now fill that in:

$$u'^{new}(x,y) = \bar{u}'^{old} - \frac{\lambda I_{x'}^2 \bar{u}'^{old} + \lambda I_{x'} 0 \bar{v}'^{old} + \lambda I_{x'} I_t}{(\lambda I_{x'}^2 + 0 + 4)} = \bar{u}'^{old} - \frac{\lambda I_{x'}^2 \bar{u}'^{old} + \lambda I_{x'} I_t}{(\lambda I_{x'}^2 + 4)} \tag{17}$$

$$u'^{new}(x,y) == \bar{u}'^{old} - \frac{I_{x'}^2 \bar{u}'^{old} + I_{x'} I_t}{(I_{x'}^2 + \frac{4}{\lambda})} \tag{18}$$

...which matches what we were seeking to reduce to for $u'^{new}(x.y)$. Doing this for $v'^{new}(x,y)$ we would follow a similar path - we would find that:

$$v'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)} \begin{bmatrix} -\lambda I_{x'} I_{y'} & \lambda I_{x'}^2 + 4 \end{bmatrix} \begin{bmatrix} 4\bar{u}'^{old} - \lambda I_{x'} I_t \\ 4\bar{v}'^{old} - \lambda I_{y'} I_t \end{bmatrix} \tag{19}$$

$$v'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + \lambda I_{y'}^2 + 4)}\big((\lambda I_{x'}^2 + 4)(4\bar{v}^{old} - \lambda I_{y'} I_t) - (\lambda I_{x'} I_{y'})(4\bar{u}^{old} - \lambda I_{x'} I_t)\big) \tag{20}$$

...again, we can utilize the specified knowledge that $I_{y'} = 0$:

$$v'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + \lambda 0^2 + 4)}\big((\lambda I_{x'}^2 + 4)(4\bar{v}^{old} - \lambda 0 I_t) - (\lambda I_{x'} 0)(4\bar{u}^{old} - \lambda I_{x'} I_t)\big) \tag{21}$$

$$v'^{new}(x,y) = \frac{1}{4(\lambda I_{x'}^2 + 4)}((\lambda I_{x'}^2 + 4)(4\bar{v}^{old})) \tag{22}$$

$$v'^{new}(x,y) = \frac{4\bar{v}^{old} \lambda I_{x'}^2 + 16\bar{v}^{old}}{4(\lambda I_{x'}^2 + 4)} \tag{23}$$

$$v'^{new}(x,y) = \bar{v}^{old} \frac{4\lambda I_{x'}^2 + 16}{4(\lambda I_{x'}^2 + 4)} \tag{24}$$

$$v'^{new}(x,y) = \bar{v}^{old} \frac{4(\lambda I_{x'}^2 + 4)}{4(\lambda I_{x'}^2 + 4)} \tag{25}$$

$$v'^{new}(x,y) = \bar{v}^{old} \tag{26}$$

...thus proving that:

$$u'^{new}(x,y) = \bar{u}'^{old} - \frac{I_{x'}^2 \bar{u}'^{old} + I_{x'} I_t}{I_x^2 + \frac{4}{\lambda}} \tag{27}$$

$$v'^{new}(x,y) = \bar{v}'^{old} \tag{28}$$

Problem 2

We can represent an object by its boundary, $(x(s), y(s)), 0 \leq s \leq S$, where S is the length of the object's boundary and s is distance along that boundary from some arbitrary starting point. Combine x and y into a single complex funcion $z(s) = x(s) + jy(s)$. The Discrete Fourier Transform (DFT) of z is:

$$Z(k) = \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} z(s), 0 \leq k \leq S-1 \tag{29}$$

We can use the coefficients $Z(k)$ to represent the object boundary. The limit on s is $S-1$ because for a closed contour $z(S) = z(0)$. The Inverse Discrete Fourier Transform is:

$$z(s) = \frac{1}{S} \sum_{k=0}^{S-1} e^{+2\pi j \frac{ks}{S}} Z(k), 0 \leq s \leq S-1 \tag{30}$$

A

Suppose that the object is translated by $(\Delta x, \Delta y)$, that is, $z'(s) = z(s) + \Delta x + j\Delta y$. How is z''s DFT $Z'(k)$ related to $Z(k)$?

$$Z(k) = \sum_{s=0}^{S-1} e^{-2*\pi j \frac{ks}{S}} z(s), 0 \leq k \leq S-1 \tag{31}$$

Which we can define $z'(s)$ as:

$$z'(s) = z(s) + \Delta x + j\Delta y \tag{32}$$

If we plug this into our original equation, we get...

$$Z'(k) = \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} z(s) + \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} (\Delta x + j\Delta y)(1) \tag{33}$$

Here we see that we have a segment that is equivalent to oru defined $Z(k)$, so we can simplify by expressing:

$$Z(k) + (1)(\Delta x + j\Delta y) \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} \tag{34}$$

With Δx and $j\Delta y$ isolated, we can use a table of known Fourier Transforms to identify the resulting conversion. Based on our problem's definition ΔX and Δy are both constants, so we can state that:

$$Z'(k) = Z(k) + (\Delta x + j\Delta y)\sigma(\frac{2\pi k}{S}) \tag{35}$$

...where $\frac{1}{S}$ acts as our scaling factor.

B

In this section, we are asked to suppose that the object is scaled by an integer constant c, that is $z'(s) = cz(s)$. For simplicity, we are to assume that $S' = S$. How is $Z'(k)$ as the DFT of z' related to $Z(k)$?
Starting with our definition of $Z(k)$, which is our Fourier Transform of $z(s)$ as defined earlier:

$$Z(k) = \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} z(s), 0 \leq k \leq S-1 \tag{36}$$

If we then define $z'(s)$ as above, $z'(s) = cz(s)$, we can plug it into our Fourier Transform:

$$Z'(k) = \sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} cz(s) \tag{37}$$

Since c is a constant, we know we can pull it in front of the fourier Transform like so:

$$Z'(k) = c\sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} z(s) \tag{38}$$

...and further simplified:

$$Z'(k) = cZ(k) \tag{39}$$

...which shows that multiplying our time series function by a given scalar simply multipies our frequncy domain by the same scalar.

C

We wish to know what object has $z(s) = \left[x_0 + R\cos(\frac{2\pi s}{s})\right] + j\left[y_0 + R\sin(\frac{2\pi s}{s})\right]$. Below we created a graph drawing the resulting shape, and include the code utilized to generate it. Arbitrary values were chosen for S, r, x_0, and y_0 for the sake of plotting.

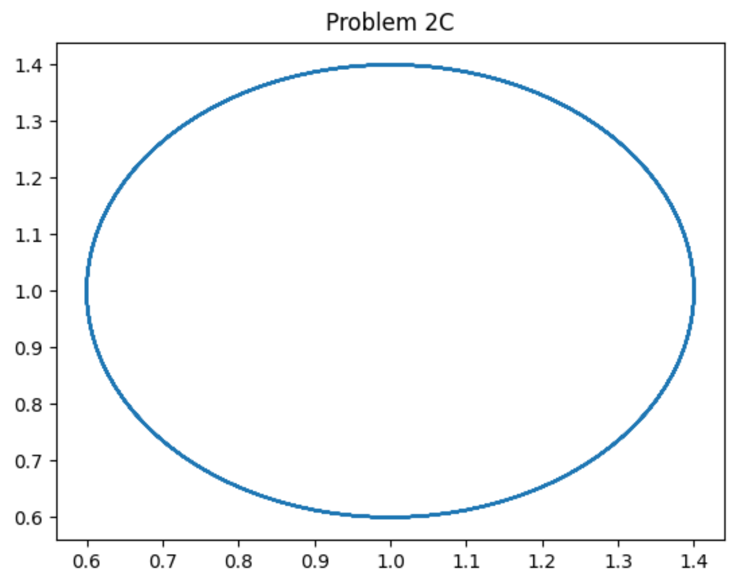

Figure 1: Our resulting shape

```
import numpy as np
from numpy import cos, sin, pi
import matplotlib.pyplot as plt

figure = plt.figure()
plt.title("Problem 2C")

S = 10
r = 4
x0 = 1
y0 = 1

theta = [theta for theta in np.arange(0, S, 0.01)]
X = [
        x0 + r * cos(2*pi*theta)/S
        for theta in theta
    ]
Y = [
        y0 + r * sin(2*pi*theta)/S
        for theta in theta
    ]

# Plot the results
plt.plot(X, Y)
plt.savefig("./imgs/prob2_c.png")
```

D

What is $Z(k)$ corresponding to $z(s)$ from Part C? To do this, we begin wtih our $z(s)$:

$$z(s) = \left[x_0 + R\cos(\tfrac{2\pi s}{s})\right] + j\left[y_0 + R\sin(\tfrac{2\pi s}{s})\right] \tag{40}$$

We can utilize the inverse of Euler's formula; $\cos(x) = \frac{e^{ix}+e^{-ix}}{2}$ and $\sin(t) = \frac{e^{ix}-e^{-ix}}{2x}$. This allows us to expand our starting equation:

$$z(s) = x_0 + R\frac{e^{j\frac{2\pi s}{S}} + e^{-j\frac{2\pi s}{S}}}{2} + jy_0 + R\frac{e^{j\frac{2\pi s}{S}} - e^{-j\frac{2\pi s}{S}}}{2} \tag{41}$$

$$z(s) = x_0 + jy_0 + \frac{R}{2}(e^{j\frac{2\pi s}{S}} + e^{-j\frac{2\pi s}{S}} + e^{j\frac{2\pi s}{S}} - e^{-j\frac{2\pi s}{S}}) \tag{42}$$

$$z(s) = x_0 + jy_0 + \frac{R}{2}(2e^{j\frac{2\pi s}{S}}) \tag{43}$$

$$z(s) = x_0 + jy_0 + Re^{j\frac{2\pi s}{S}} \tag{44}$$

The Discrete Fourier Transform (DFT) from earlier in our problem we can begin to expand this equation now. Starting with:

$$Z(k) = \sum_{s=0}^{S-1} e^{-2\pi j\frac{ks}{S}} z(s), 0 \leq k \leq S-1 \tag{45}$$

...which can lead us to:

$$Z(k) = \sum_{s=0}^{S-1} e^{-2\pi j\frac{ks}{S}} \left(x_0 + jy_0 + Re^{j\frac{2\pi s}{S}}\right) \tag{46}$$

...constants can be pulled out, leaving us with:

$$Z(k) = \sum_{s=0}^{S-1} e^{-2\pi j\frac{ks}{S}} (x_0 + jy_0) + R\sum_{s=0}^{S-1} e^{-2\pi j\frac{ks}{S}} e^{j\frac{2\pi s}{S}} \tag{47}$$

$$Z(k) = (x_0 + jy_0)\sum_{s=0}^{S-1} e^{-2\pi j \frac{ks}{S}} + R\sum_{s=0}^{S-1} e^{-j2\pi(k-1)}S \tag{48}$$

We know from a Fourier Transform lookup table we can then convert this to:

$$Z(k) = (x_0 + jy_0)\sigma(\frac{2\pi k}{S}) + \sigma(2\pi\frac{k-1}{S}) \tag{49}$$

Problem 3

In this problem, we are looking at Stereo vision via Singular Value Decomposition (SVD). We are told to assume the usual stereo geometry, where the left and right cameras are offset by baseline $\vec{B}$ that is perpendicular to the common focal vector $\vec{F}$. Then the stereo imaging equations are:

$$\vec{X}_L = \frac{|\vec{F}|^2}{\vec{F}\cdot\vec{X}^W} \tag{50}$$

$$\vec{X}_R = \frac{|\vec{F}|^2}{\vec{F}\cdot\vec{X}^W} \tag{51}$$

In the presence of imaging errors and noise, these equations might not hold exactly. We can approximate them by:

$$\vec{X}_L - \frac{|\vec{F}|^2}{\vec{F}\cdot\vec{X}^W}(\vec{X}^W + \frac{\vec{B}}{2}) \approx \vec{0} \tag{52}$$

$$\vec{X}_R - \frac{|\vec{F}|^2}{\vec{F}\cdot\vec{X}^W}(\vec{X}^W + \frac{\vec{B}}{2}) \approx \vec{0} \tag{53}$$

A

First we are tasked with showing that these equations can be written as a $4x4$ matrix operating on a column vector in homogenous coordinates:

$$\begin{bmatrix} -f & 0 & x_L & -f\frac{b}{2} \\ 0 & -f & y_L & 0 \\ -f & 0 & x_R & -f\frac{b}{2} \\ 0 & -f & y_R & 0 \end{bmatrix} \begin{bmatrix} x^W \\ y^W \\ z^W \\ 1 \end{bmatrix} \approx \vec{0} \tag{54}$$

We start by first rewriting the project equation for the left camera:

$$(\vec{F}\cdot\vec{X}^W)\vec{X}_L - |\vec{F}|^w(\vec{X}^W + \frac{\vec{B}}{2}) \approx \vec{0} \tag{55}$$

...we can then expand the components for each coordinate axis:

$$fz^W x_L - f^2 x^W - f^2\frac{b}{2} \approx 0 \tag{56}$$

$$fz^W y_L - f^2 y^W \approx 0 \tag{57}$$

$$z^W y_R - fy^W \approx 0 \tag{58}$$

...where the bottom equation reduces to 0, as $z_L = f$. We can then simplify these equations for both the left and right projections by dividing them by f to simplify:

$$z^W x_L - fx^W - f\frac{b}{2} \approx 0 \tag{59}$$

$$z^W y_L - fy^W \approx 0 \tag{60}$$

$$z^W x_R - f x^W + f\frac{b}{2} \approx 0 \tag{61}$$

$$z^W y_R - f y^W \approx 0 \tag{62}$$

...We can represent this as a matrix:

$$\begin{bmatrix} -f & 0 & x_L & -f\frac{b}{2} \\ 0 & -f & y_L & 0 \\ -f & 0 & x_R & -f\frac{b}{2} \\ 0 & -f & y_R & 0 \end{bmatrix} \begin{bmatrix} x^W \\ y^W \\ z^W \\ 1 \end{bmatrix} \approx \vec{0} \tag{63}$$

...which is what we were looking for!

B

We can use SVD to find the singular vector $\tilde{X}'$ that minimizes $|A\vec{X}|^2$ subject to $|\vec{X}|^2 = 1$. We wish to express the world point $\vec{X} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$ in terms of $\tilde{X}' = \begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix}$.

For this we can simply say that:

$$\vec{X}^W = \frac{1}{W'}\tilde{X}' = \begin{bmatrix} \frac{x'}{w'} \\ \frac{y'}{w'} \\ \frac{z'}{w'} \\ 1 \end{bmatrix} \tag{64}$$

C

Here we are tasked to show that as $y_L = y_R$, our answer in part A gives $z^W = \frac{fb}{d}$, where d is the disparity.
To start, we look at our equations from part A:

$$z^W x_L - f x^W - f\frac{b}{2} \approx 0 \tag{65}$$

$$z^W y_L - f y^W \approx 0 \tag{66}$$

$$z^W x_R - f x^W + f\frac{b}{2} \approx 0 \tag{67}$$

$$z^W y_R - f y^W \approx 0 \tag{68}$$

Looking at the above equations, we can simplify the 2nd and 4th equations; they simplify to $z^W y - f y^W \approx 0$. If we set $y^W = \frac{z^W}{f}$, then it equals 0 and can be ignored. Then, looking at equations 1 and 3 and subtracting one of the other:

$$(x_L - x_R) z^W 0 f b \approx 0 \tag{69}$$

...which expressed differently, shows that:

$$z^W = \frac{fb}{x_L - x_R} \tag{70}$$

...and since we defined $d = x_L - x_R$...:

$$z^W = \frac{fb}{d} \tag{71}$$

...which is what we set out to prove!

Problem 4

In this problem we are presented with a robot which has two main sensors for vision - an infra-red sensor, and a LiDAR sensor.

We are tasked with filling in a table with brief discussions of one to three challenges with LiDAR vision alone, infra-red vision alone, the combination of the two, and challening situations that require additional vision capacities.

	Challenges w/ LiDAR	Challenges w/ infra-red	Benefits w/ both	Other vision options?
# 1	LiDAR can have difficulty differentiating objects clustered together as separate objects, as their surfaces present pointclouds that merge together.	Infra-red alone has trouble differentiating objects grouped together if their materials are of similar colors. Since we're seeing the resulting heat emission of all objects, if they are emitting similar temperatures they will look as a singular clump.	In both you can...	