10.10.2023

Sygnały i Obrazy Cyfrowe

Sprawozdanie 1 - Metody próbkowania sygnałów i obrazów - Aliasing 2D

Mateusz Marko, grupa nr 1, TN wt 17:05
Nr 273168, ISA

1. Cel ćwiczenia

Zbadanie zjawiska aliasingu przez użycie sensora o odczycie sekwencyjnym oraz zmianę obrazu w zależności od różnych parametrów funkcji dla ruchomych obiektów dla przykładu na obracającym się śmigle.

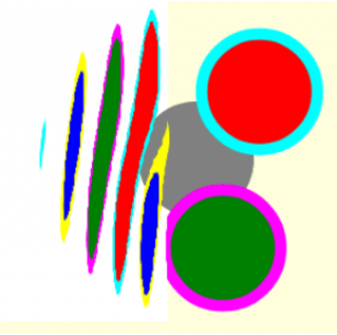
2. Wstęp teoretyczny

- ## Aliasing

Zjawisko, które występuje, gdy próbkowanie sygnału (lub obrazu) jest wykonywane z zbyt niską częstotliwością w stosunku do częstotliwości sygnału. Wynikiem tego jest błędne odtworzenie faktycznego obrazu, co prowadzi do zniekształceń i błędów w reprezentacji sygnału lub obrazu. W przypadku nagrywania kręcącego się śmigła trójłopatowego samolotu, aliasing może wystąpić, gdy kamera lub sensor nagrywający obraz śmigła wykonuje zbyt niskie próbkowanie w porównaniu z szybkością obrotu śmigła. To zjawisko może prowadzić do efektów takich jak zniekształcenia (Jeśli częstotliwość obrotu śmigła jest zbyt wysoka w porównaniu z częstotliwością próbkowania kamery, to śmigło może wydawać się obracać w przeciwnym kierunku lub z inną prędkością niż w rzeczywistości), czy efekt migotania (łopatki śmigła wydają się migotać lub poruszać w sposób nieregularny na nagraniu). Aby uniknąć aliasingu wystarczy po prostu zwiększyć częstotliwość próbkowania kamery/sensora. Dla dokładnych pomiarów prędkości obrotu i uniknięcia zniekształceń ważne jest, aby zapewnić odpowiednią częstotliwość próbkowania w stosunku do prędkości obrotu obiektu, który jest nagrywany.

- ## Rolling shutter (przesuwająca się migawka)

Sposób, w jaki niektóre aparaty cyfrowe oraz kamery wideo pobierają obraz. W przeciwności do tradycyjnych migawek, które otwierają się jednocześnie, migawka przesuwająca się otwiera się i zamyka sekwencyjnie, przesuwając się od góry do dołu (lub od dołu do góry) w stosunku do obrazu. Gdy migawka przesuwająca się zaczyna zbierać obraz, to oznacza, że nie wszystkie elementy obrazu są pobierane jednocześnie. Zamiast tego migawka przesuwa się po obrazie, zbierając informacje o każdej części obrazu sekwencyjnie czyli na przykład po kolei pobierając linie z góry do dołu. W przypadku śmigła, kiedy łopatki kręcą się szybko, mogą wystąpić zniekształcenia. Dla części migawki, która jest bliżej górnej części obrazu, łopatki mogą być w innym położeniu niż dla części bliżej dolnej. W efekcie łopatki mogą wydawać się wygięte lub nieprawidłowo umieszczone na obrazie. To zjawisko jest szczególnie widoczne przy szybko obracających się obiektach, takich jak właśnie śmigło, fidget spinnery czy koła, ponieważ każda część obrazu jest rejestrowana w inny sposób w zależności od czasu. Aby zminimalizować wpływ rolling shutter na obrazy, można stosować krótsze czasy migawki lub zmniejszać prędkość obrotu obiektu wtedy nie będzie aż takich zniekształceń. (Co ciekawe istnieją kamery z global shutter, gdzie cały obraz jest pobierany jednocześnie)



Rysunek 1 Przykładowe przedstawienia zniekształcenia obrazu na fidget spinnerze

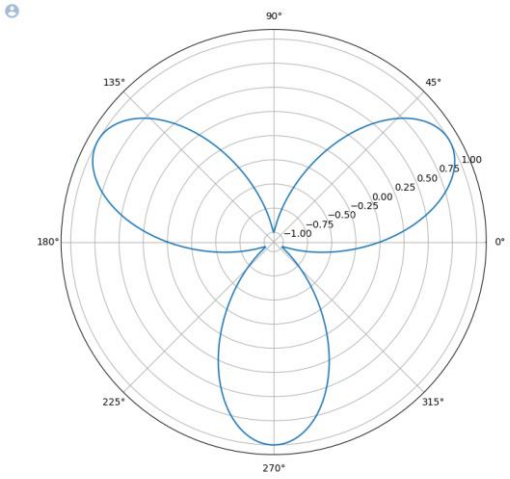
3. Przebieg ćwiczenia

W zadaniu należało wygenerować 64 obrazową animację obracającego się trój-łopatkowego śmigła (potem także pięcio-łopatkowego śmigła) z pomocą poniższego wykresu funkcji używając Pythona. A następnie sprawdzić rezultat działania sensora sekwencyjnego po zmianie konkretnych parametrów.

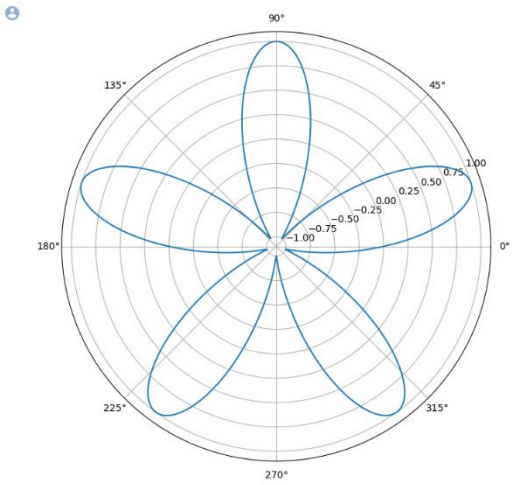
$$f(x) = \sin\left(3x + \frac{m\pi}{10}\right), \qquad M = 64, \qquad m = -\frac{M}{2}, \dots, \frac{M}{2}$$

Pięcio-łopatkowe śmigło

$$f(x) = \sin\left(5x + \frac{m\pi}{10}\right), \qquad M = 64, \qquad m = -\frac{M}{2}, \dots, \frac{M}{2}$$



Rysunek 2 Zrzut ekranu obracającego się śmigła trój-łopatowego w google colab



Rysunek 3 Zrzut ekranu obracającego się śmigła pięcio-łopatowego w gogle colab

```python
def propeller(theta, m):
    return np.sin(N*theta + m * np.pi /20)
```

```python
thetas = np.linspace(-np.pi, np.pi, 1000)
r = propeller(thetas, m=0)

_ = plt.figure(figsize=[8, 8])
_ = plt.polar(thetas, r)
```

Rysunek 4 Kod odpowiadający za projekcje śmigła

```python
def animate(frame):
    propeller_curve = propeller(thetas, m=frame)
    plot.set_data((thetas, propeller_curve))
```

```python
animation = FuncAnimation(figure, animate, frames=100, interval=25)
```

```python
video = animation.to_html5_video()
html = display.HTML(video)
display.display(html)

plt.close()
```

Rysunek 5 Kod odpowiadający za animacje obrotu śmigła

Zmieniając kolejno parametry funkcji uzyskamy różne rezultaty obrazu przez to efekt końcowy będzie się nieco różnić. Dla przykładu wraz ze wzrostem a wewnątrz funkcji animacja obrotu będzie coraz wolniejsza, natomiast zmiana wartości n zmieni ilość śmigieł.

$$f(x) = \sin\left(nx + \frac{m\pi}{a}\right)$$

Przy tak dowolnie dobieranych parametrach przy użyciu poniższego kodu jesteśmy w stanie zaobserwować różne rezultaty zjawiska sekwencyjnego odczytywania obrazu

```
funcs = []

for m in ms.tolist():
    r = propeller(thetas, m=m)
    func = polar_to_cartesian(thetas, r)
    funcs.append(func)

funcs = np.asarray(funcs)
```

```
def record(funcs: list, capture_kwargs) -> np.array:
    """Simulate recording by applying capture in loop"""
    return np.asarray([capture(func, **capture_kwargs) for func in progress_bar(funcs)])
```

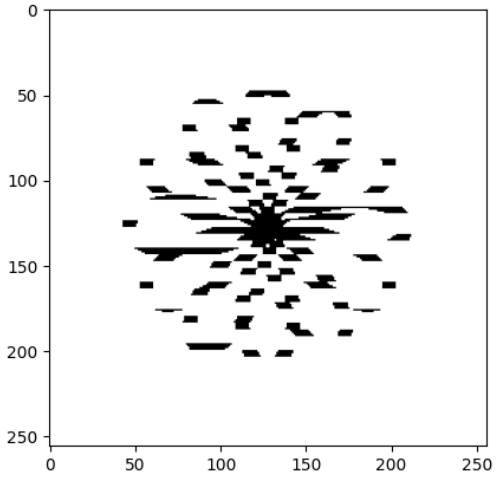
```
recording = record(funcs, capture_kwargs=dict(resolution=256, threshold=0.05, low=-np.pi / 2, high=np.pi / 2))
recording.shape
```

```
offset = 0
length = 4
capture = np.zeros([256, 256])
for frame in recording:
    capture[offset : offset + length, :] = frame[offset : offset + length, :]
    offset += length
```

```
plt.imshow(capture, cmap="Greys")
```

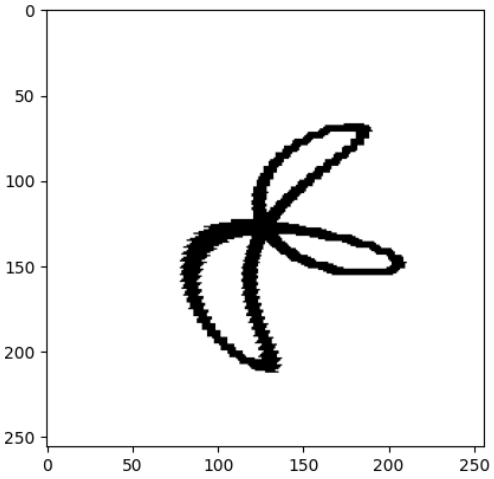
Rysunek 6 Kod odpowiadający za działania sensora o odczycie sekwencyjnym

Oto parę przykładów generowanych obrazów dla konkretnych parametrów:



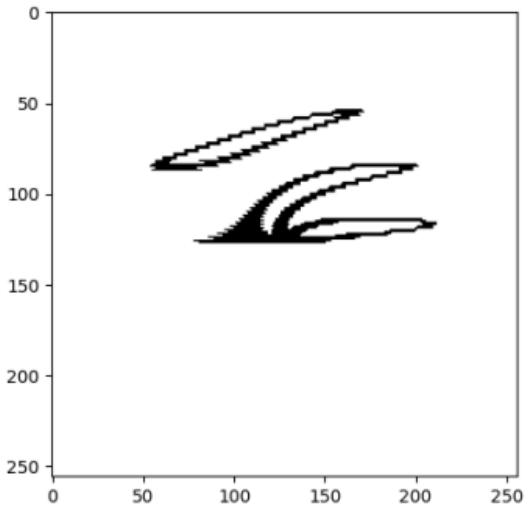

```
def propeller(theta, m):
    return np.sin(N*theta + m * np.pi /2)
```

Rysunek 7 Odczyt po zmianie prędkości śmigła na 5 razy szybszą

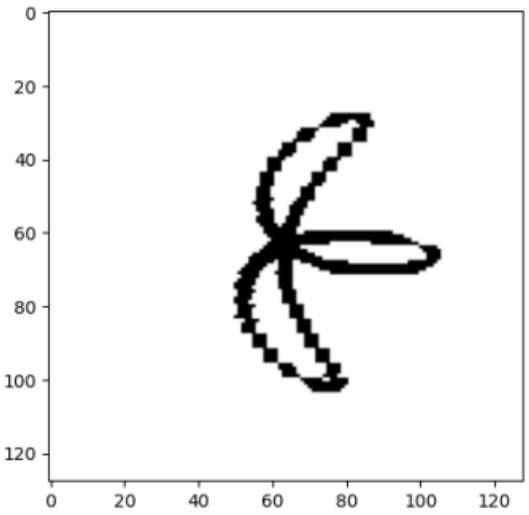



```
def propeller(theta, m):
    return np.sin(N*theta + m * np.pi /20)
```

Rysunek 8 Odczyt po zmianie prędkości śmigła na 2 razy wolniejszą



Rysunek 9 Odczyt po zmianie length (długości) na 2



Rysunek 10 Odczyt po zmianie resolution (rozdzielczości) na 128

4. Wnioski

- Zakłócenia w generowanych obrazach wynikają przede wszystkim z Aliasingu oraz z niespełnienia twierdzenia o próbkowaniu, które mówi iż częstotliwość próbkowania powinna być co najmniej dwukrotnie większa niż najwyższa częstotliwość w sygnale wejściowym.
- Aby zredukować te zakłócenia oraz uniknąć zniekształceń, można po prostu zwiększyć częstotliwość wyświetlania obrazu. Wystarczy jedynie dostosować częstotliwość zgodnie z powyższym twierdzeniem. W naszym przypadku gdzie generujemy obrazy przedstawiające kręcące się śmigło, może to pomóc w zmniejszeniu występowania zakłóceń i nietypowych zniekształceń.
- W przypadku tworzenia uniwersalnej funkcji, która pozwala na wyświetlanie dowolnej ilości łopatek, można zmienić parametr "n" w funkcji sinusoidalnej. Zmieniając "n", zmienia się liczbę okresów funkcji, co pozwala na dostosowanie liczby łopatek. Przykład:

$$f(x) = \sin\left(nx + \frac{m\pi}{10}\right),$$