R の基礎と画像データ処理

2024/04/22

HI4 45 号 山口惺司

1. 実験目的

動画を創作し動画の原理と制作技術の基礎を習得する。また、R の基本的な使い方ができ、画像データを取り扱うことができる。

2. 実験項目

1. R の基本的な使い方と画像処理用パッケージのインストール
2. 動画制作
3. R の画像処理パッケージを用いた画像データの処理

3. 課題内容

1. R の基本的な使い方と画像処理用パッケージのインストール

   課題 1-1 R の基本的な使い方について、以下を実施する。

   (a). ベクトルおよびデータフレームの使用例について、スクリプト、実行結果を示し、説明する。

   (b). 自作関数の実行例について、スクリプト、実行結果を示し、説明する。

   課題 1-2 画像処理用パッケージのインストールと動作確認について、以下を実施する。

   (a). 各自の PC にパッケージ imager をインストールする。

   (b). 以下のスクリプトを実行して、画像データが表示されることを、各自の PC を用いてデモを見せる。

   R スクリプト

   ```
   library(imager)
   plot(boats)
   ```

2. 動画制作

   課題 2 複数の静止画像をフレームデータ(ビデオの 1 画像)として、時系列に並べて動画を制作する。

   ・フレームデータを動画制作ソフトに取り込み、10 秒以内で動画を制作する。このとき、必要に応じて新たなフレーム画像を作成し、加除修正する。

   ・完成した動画のフレーム周期（フレームレート）を変化して見え方（動きの滑らかさや面白さなど）を各自で評価する。

   ・制作した作品は、動画ファイル（.mp4）にして保存する。このときファイル名は、「出席番号・氏名－課題 2」とすること。

3. R の画像処理用パッケージを用いた画像データの処理

   課題 3

   ・課題 2 で利用したフレーム(ビデオの 1 画像)データを R に取り込み、画像処理パッケージを利用した処理を実施する。

   ・処理する画像は 3 枚

   ・なお、実施する処理は、各自の興味あるものをする。

4. 課題 2 の考察

課題 2 の動画制作では「Krita」というソフトウェアを使用して動画を制作した。
10fps で 10 秒つまり、100 枚の画像で動画を制作した。
工夫した点はリアリティを再現した点である。
８枚の画像を図 1 に示す。
一見、同じ画像に見えるが、全部微妙に変わっている。
これは、スナイパーがスコープを覗いた時の手振れを再現したものになる。

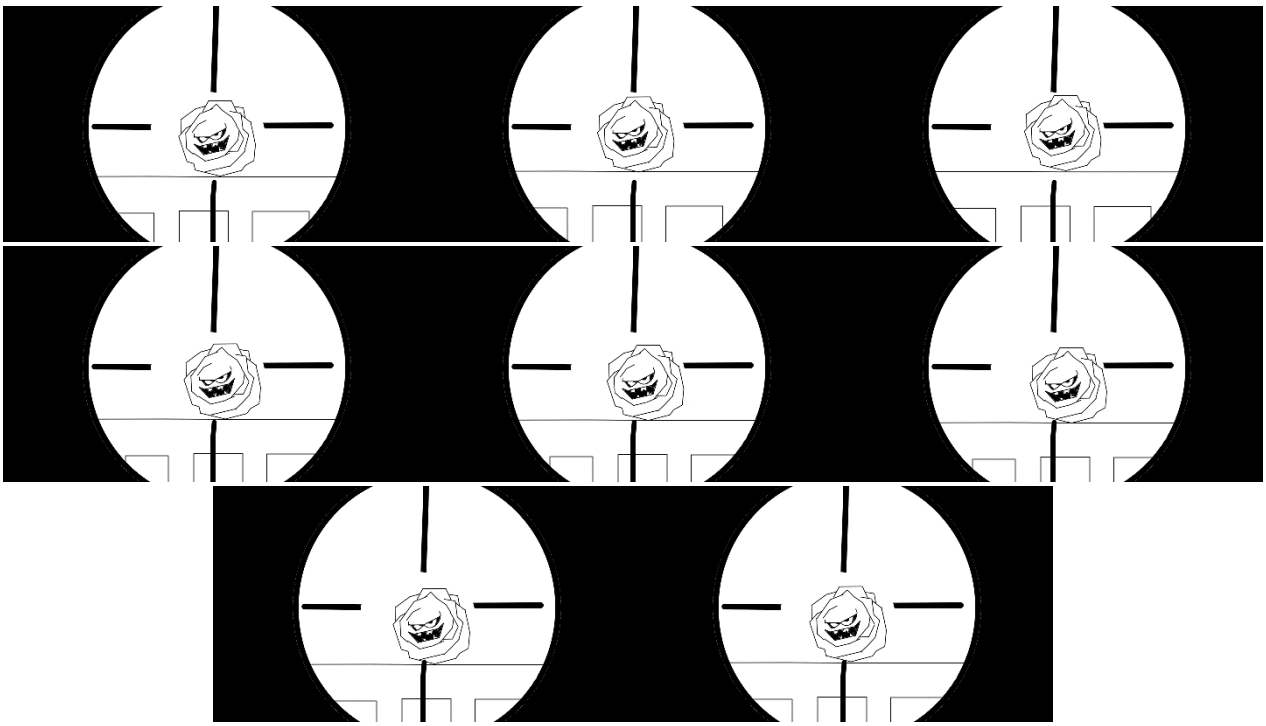
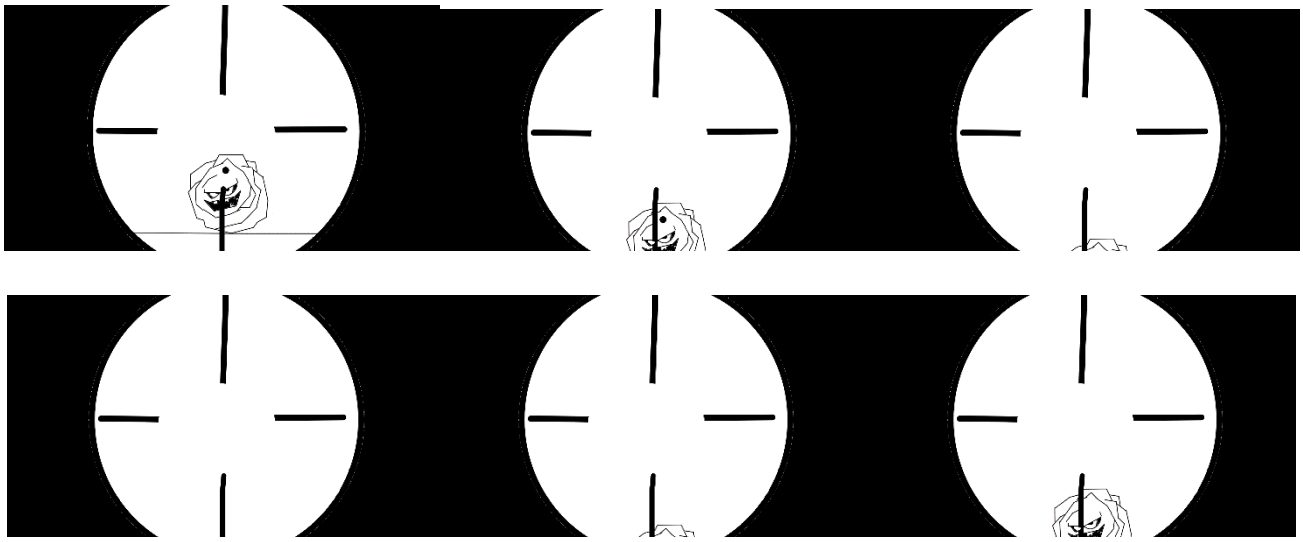


図 1 53～61 フレーム、手振れを再現した画像

また、銃を撃った時の反動を再現したフレームを図 2 に示す。



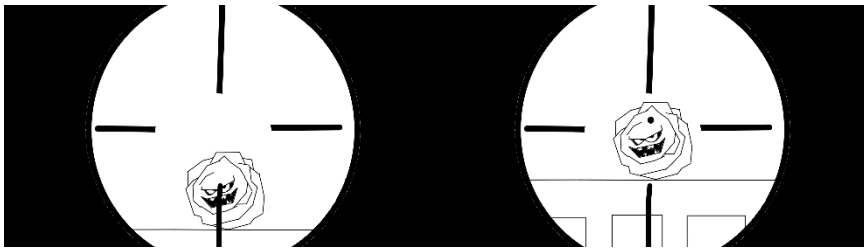


図 2　72～79 フレーム目、銃を撃った時の反動の再現

5. 課題 3 の考察

ソースコードを以下に示す。

```
1       kadai3 <- function(){
2       library(imager)
3
4       img1 <- load.image("imgs/53.png")
5       img1 <- mirror(img1, "y")
6       dev.new()
7       plot(img1)
8
9       img2 <- load.image("imgs/36.png")
10      dev.new()
11      img2 <- isoblur(img2, 10) %>% plot(main="Isotropic blur, sigma=10")
12
13      img3 <- load.image("imgs/37.png")
14      img3 <- deriche(img3, 2, order=2, axis="x")
15      dev.new()
16      plot(img3)
17      }
```

図 3 画像処理 R スクリプト

では、それぞれの画像処理について解説する。

まず、img1 について、53 フレーム目の画像処理前の画像を図 5、画像処理後の画像を図 6 に示す。

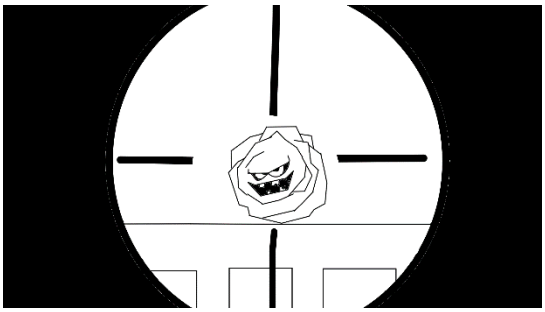


図 4 フレーム 53 画像処理前

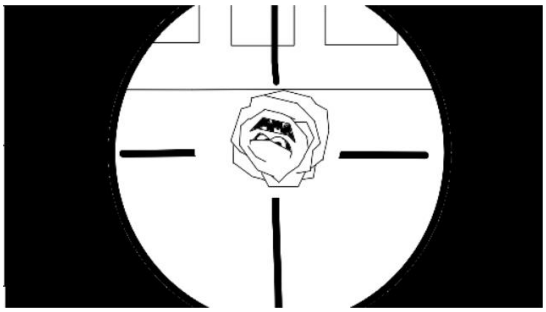


図 5 フレーム 53 画像処理後

これは図 3、5 行目の mirror 関数を使用して上下を反転させている。

mirror 関数の使い方は以下の通りである。

mirror(im, axis)

im は画像の引数で axis は反軸の引数である。axis の値を y から x に変えると、左右の反転もできる。

次に、img2 について、36 フレーム目の画像処理前の画像を図 7、画像処理後の画像を図 8 に示す。

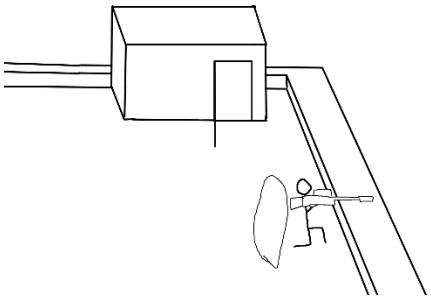


図 6 フレーム 36 画像処理前

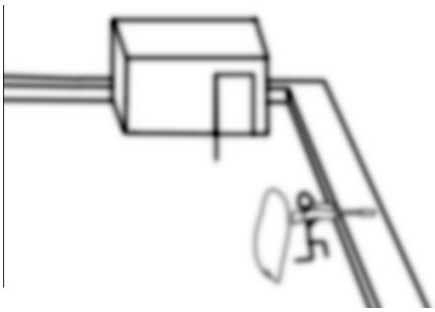


図 7 フレーム 36 画像処理後

これは図 3、11 行目の isoblur 関数を使用して、画像をボケさせている。
isoblur 関数の使い方は以下の通りである。

isoblur(im, sigma)

im は画像の引数、sigma はボケの標準偏差である。sigma の値を大きくするとボケが大きくなる。

最後に、img3 について、37 フレーム目の画像処理前の画像を図 9、画像処理後の画像を図 10 に示す。

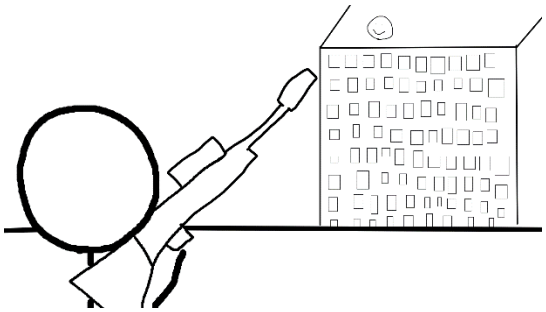


図 8 フレーム 37 画像処理前

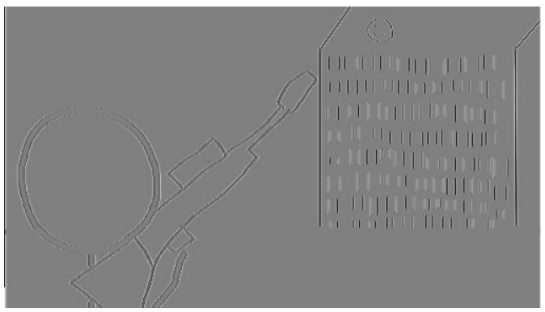


図 9 フレーム 37 画像処理後

これは図 4、14 行目の deriche 関数を使用してガウシアンフィルタをかけている。
deriche 関数の使い方は以下の通りである。

deriche(im, sigma, order, axis)

im は画像の引数、sigma はフィルタの標準偏差、order は 0, 1, 2 の値が入り、0 はスムージングフィルタ、1,2 は 1 次微分,2 次微分によるフィルタ、axis は軸の引数である。

6. 研究事項

フレームレートについて

フレームレートとは 1 秒間に表示される静止画の数を表している。
今回の実験で制作した動画は 10fps だが、普段我々が使用しているテレビやゲーム、映画などのフレームレートはいくつぐらいなのか気になったので調べた。

まず、映画は一般的に 24fps で作られている。映画のフレームレートは 1920 年代に決定された標準で、映画的な質感を与える効果がある。また、もし 60fps で映画を作った場合、安っぽく見えるらしい。
次に、テレビの映像、これは一般的に 30fps で作られている。
そしてゲームでは一般的に 30~240fps ほどを使用する。なぜこれだけばらつきがあるかというと、ゲームによって要求されるフレームレートが違うからだ、ほとんど動きのないゲームだと低いフレームレートでも問題ないが、より高速なアクションを求められるゲームだとフレームレートが高くないと何をやっているかわからない。
また、フレームレートが高ければ高いほど PC が処理をするのに時間がかかるため、同じゲームでも人によって使用するフレームレートが違うこともある。

7. 感想

・今回 10fps で 10 秒の動画を制作したが、24fps で 2 時間の映画を作るアニメ関係の仕事に就いている人はとてもすごいと感じた。
・3 年生の時に Python で二値化や画像の反転のプログラムを書いたが、R の imager ライブラリを使用することでプログラムを書く手間が省けることに感動した。

8. 参考文献

https://educon.jp/knowledge/10242/#:~:text=24fps%3A%20%E6%98%A0%E7%94%BB%E3%81%A7%E4%B8%80%E8%88%AC%E7%9A%84,%E3%81%95%E3%82%8C%E3%82%8B%E3%83%95%E3%83%AC%E3%83%BC%E3%83%A0%E3%83%AC%E3%83%BC%E3%83%88%E3%81%A7%E3%81%99%E3%80%82