Introdution to TubAR

Michael D Miller, Cari Schmitz Carley, Laura M Shannon

7/20/2021

Abstract

TubAR (Tuber Analysis in R) provides potato researchers with a simple means of quantitative phenotyping replacing previously used imprecise qualitative scales. Users collect images of sample tubers using a light box, and then use the package to quantify: tuber length, width, roundness, skinning percentage, redness, and lightness. Finally, the data is prepared for export and use in mapping, genomic selection, etc.

Installing the package

Use the `install_github` function from the devtools package to install TubAR. TubAR makes use of the EBImage and Biobase packages, which are available through BiocManager, not CRAN, so you will also have to install those packages before TubAR.

```r
install.packages("devtools")
install.packages("BiocManager")
BiocManager::install("EBImage")
BiocManager::install("Biobase")
devtools::install_github("michaelmiller45/TubAR")
```

Collecting Image Data

TubAR is designed for use with a Photosimile 200 lightbox, however other lightboxes with approximately a 60cm by 60cm base should be compatible. The authors used a Canon Rebel T6i camera using a 24mm lens, ISO 100, 1/30 sec shutter speed and aperture f/5.6 to take photos. Potatoes should be washed and dried before imaging for accurate results. TubAR is intended for evaluating up to 10 tubers per photo. With large potatoes, fewer tubers per photo may be necessary to avoid contact between tubers. All tubers in one photo should be the same clone. A 24ColorCard 2x3 from CameraTrax should be included in a corner of the camera view for color correction (This specific color card is necessary to ensure proper color correction). Ensure this card remains straight, with the text at the bottom of the card, as shown in example pictures. If the card is crooked, this may hinder color correction. You may find it helpful to include a tag that identifies the pictured clone. Bright red or bright green, 8cm by 12cm rectangular tags are recommended for reliable removal of tags from data. Setting the tag in the upper left corner with no space between the tag and the edge of the picture is ideal. Input images should be in a JPEG format. Image file names should be chosen so that it is easy to identify the clone pictured without having to open and look at the photo. The zoom and camera placement should remain consistent for all samples that are to be compared. See Caraza-Harter & Endelman (2020) for further information on set up.

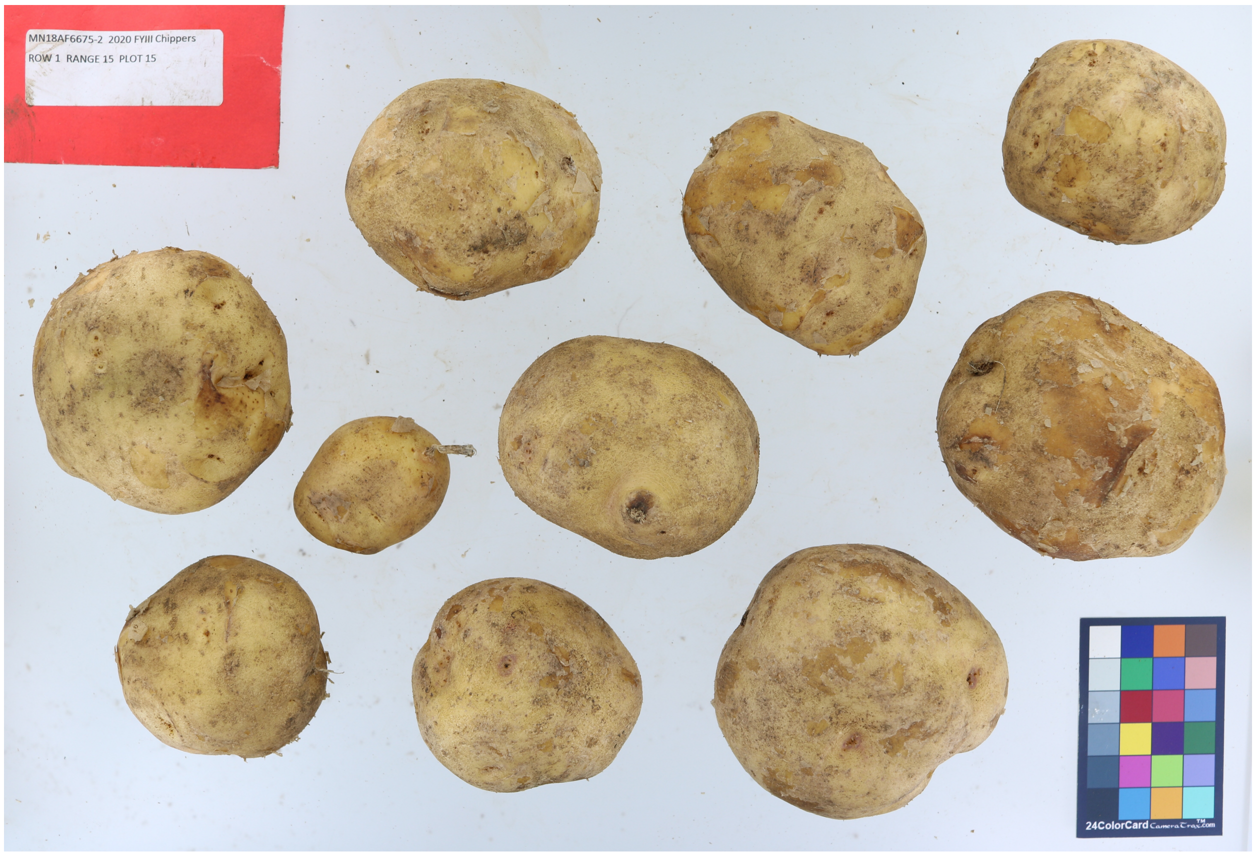

Figure 1: This is an example of a photo to be imput into TubAR. Notice color card placement in the bottom right, the tag placement in the upper left, and tuber spacing.

Example Walkthrough

We will show the data collection and organization workflow using an example data set of 11 photos.

Photos were taken using the previously described means. Photos are housed in a single file that was then set as the working directory.

```
library(TubAR)
setwd("./inst/images")
```

The `shape.all()` and `skin.all()` functions were used to create the `shape_exmp` and `skin_exmp` data. `skin.all()` will likely take several minutes.

```
shape_exmp <- shape.all(n.core=1)
skin_exmp <- skin.all(n.core=1, display=T, mode="debug", write.clean=F, pix.min=4e3, scaledown=4, color
```

The `shape.all()` output `shape_exmp` is formatted as an array with a row for each potato. Each row lists the image the potato is from along with the measurements for each trait. No median value per image is given in this data form.

The `skin.all()` output `skin_exmp` is formatted as a list with two elements `median.values` and `by.tuber`. `median.values` is an array of the median values for each trait for each photo. `by.tuber`consists of sublists for each image which contain lists of individual tuber values for each trait.

Below are small portions of the data to show how the data is formatted at this stage.

```
shape_exmp[1:2,]
```

```
##      image         bbox.width         bbox.height        perim
## [1,] "2020fy2_092" "321"              "350"              "1269"
## [2,] "2020fy2_092" "271.15482879062" "297.833747544617" "1057"
##      convex.perim      area    chull.area roundness
## [1,] "1077.0093244586" "89659" "90844"    "0.984163345514088"
## [2,] "905.87935449058" "63414" "64369"    "0.985702600570451"
##      compactness         max.length
## [1,] "0.699649471710274" "356.866921975125"
## [2,] "0.713255349648084" "302.654919008431"
```

```
skin_exmp$median.values
```

```
##                  %skinned redness lightness
## 2020fy2_092          0.00    7.50     66.30
## 2020fy2_317          0.00   14.85     54.40
## FY3Ch1_15_15         0.00   13.00     57.30
## FY3Ch5_4_68          0.00   14.20     57.40
## FY3NCRT25_21_22      0.04    6.10     18.35
## FY3NCRT26_20_61      0.00   20.55     48.15
## FY3Rd19_14_21        0.00   22.80     34.10
## FY3Rd19_4_31         0.00   26.70     33.60
## FY3Rd19_9_26         0.00   26.35     31.90
## FY3Ru10_2_7          0.00   17.40     50.40
## FY3Ru12_8_7          0.00   17.90     47.90
```

```r
skin_exmp$by.tuber[1]
```

```
## $`2020fy2_092`
## $`2020fy2_092`$skinning
##  [1] 0 0 0 0 0 0 0 0 0 0
##
## $`2020fy2_092`$redness
##  [1] 8.0 7.3 5.4 6.1 8.6 6.7 8.8 8.4 6.9 7.7
##
## $`2020fy2_092`$lightness
##  [1] 66.1 66.3 68.0 69.2 64.4 68.7 64.4 65.5 67.1 66.3
```

In order to summarize and better format the data for export to other programs or packages, as well as remove tags, one more step is required, using the `skin.export()` and `shape.export()` functions.

```r
setwd("./inst/images")
shape.export(shape_exmp, remove_tag=T)
```

```
##                   bbox.width bbox.height  perim convex.perim    area chull.area
## 2020fy2_092         263.4602    272.0247 1046.0     884.1216 59266.0   60671.75
## 2020fy2_317         183.2385    306.6609  965.0     802.9267 44906.0   45816.50
## FY3Ch1_15_15        252.8857    301.1254 1046.5     883.3761 59289.0   60355.00
## FY3Ch5_4_68         228.1120    247.6184  910.0     763.2888 45137.0   45956.75
## FY3NCRT25_21_22     175.5659    206.2055  724.0     617.3096 29018.5   29655.75
## FY3NCRT26_20_61     205.1755    246.2289  841.5     706.9490 37962.0   38856.25
## FY3Rd19_14_21       154.8855    184.1521  648.0     545.5282 22546.0   23152.50
## FY3Rd19_4_31        208.9680    247.0470  853.0     730.7487 41750.0   42534.00
## FY3Rd19_9_26        204.5612    243.1157  896.5     761.2381 43560.5   44553.25
## FY3Ru10_2_7         221.3999    365.9434 1135.5     948.5782 63195.0   64419.50
## FY3Ru12_8_7         212.7755    428.3715 1215.5    1057.4135 72577.5   73768.25
##                   roundness compactness max.length
## 2020fy2_092       0.9849330   0.6993487   297.0992
## 2020fy2_317       0.8884139   0.5943434   309.8324
## FY3Ch1_15_15      0.9818411   0.6882846   305.6023
## FY3Ch5_4_68       0.9889982   0.6810883   257.3526
## FY3NCRT25_21_22   0.9837687   0.6935290   212.3176
## FY3NCRT26_20_61   0.9760261   0.6941603   246.8082
## FY3Rd19_14_21     0.9645858   0.6745560   200.4121
## FY3Rd19_4_31      0.9699363   0.6897906   255.1917
## FY3Rd19_9_26      0.9694421   0.6770761   264.7686
## FY3Ru10_2_7       0.8898446   0.6432879   367.2716
## FY3Ru12_8_7       0.8277488   0.6260128   431.2712
```

```r
skin.export(skin_exmp, remove_tag=T)
```

```
##                   redness skinning lightness
## 2020fy2_092         7.500     0.00    66.300
## 2020fy2_317        14.850     0.00    54.400
## FY3Ch1_15_15       13.025     0.00    57.400
## FY3Ch5_4_68        14.225     0.00    57.025
## FY3NCRT25_21_22     6.100     0.04    18.350
## FY3NCRT26_20_61  20.550     0.00    48.150
## FY3Rd19_14_21    22.400     0.00    34.050
## FY3Rd19_4_31     27.750     0.00    33.750
## FY3Rd19_9_26     26.350     0.00    31.900
## FY3Ru10_2_7      17.400     0.00    50.500
## FY3Ru12_8_7      17.900     0.00    48.125
```

These functions give two matrices of the median values of each trait for each image. In order to ensure these functions work correctly the working directory should be set to the same folder the images are in.

Writing a csv file of median trait values is often a simple way of preparing the output data for use in other programs and databases. If this is desired, the `write.csv` function can create a csv file for each matrix in the current working directory.

```r
write.csv(shape_exmp, file= "shape_exmp.csv", quote = FALSE)
write.csv(skin_exmp, file= "skin_exmp.csv", quote = FALSE)
```

Functions Overview

TubAR trait measurement is divided between two groups of functions: shape and skin functions. Shape functions measure traits using the shape and dimensions of the potato perimeter while skin functions measure traits using the color of the potato surface.

Shape Fuctions

TubAR can measure potato shape in eight ways using `find.shape()`. Multiple photos can be processed as a batch by using `shape.all()`.

Function Inputs and Arguments

`find.shape()` takes an input JPEG image as well as several arguments to optimize speed and ensure potatoes are separated from the colorcard and debris.

- `pix min =` Sets the minimum pixel size of an object to not be removed with the background. This helps to remove dirt, flaked skin, or other debris that the function might otherwise attempt to measure as a potato. By default, this is set at !!!4e3!!!. If the function is measuring too many shapes per photo this can be increased (An order of magnitude increase is reasonable in trying to calibrate this). On the other hand, if small potatoes are being excluded the number can be lowered.
- `scaledown =` Sets the degree by which the image will be reduced in size to speed up processing. By default this is 8. Increasing the number will potentially speed up processing, however loss of detail will eventually lead to inaccurate measurements or errors.
- `colorcard =` Lists which corner the colorcard is positioned in. This is set to “bottomright” by default. “bottomleft”, “topright”, and “topleft” are the other settings.

If processing many photos, `shape.all()` will process all photos in the working directory. Keep the number of photos in the directory folder below 30 photos when using RStudio to avoid errors from sending too much data into the function. Running R in the command line can allow for larger photo batches.

- `n.core =`: Determines the number of processor cores to use in the function. This is set to 1 by default. This setting is going to depend on your computer and is usually limited if running R through RStudio.

Trait outputs

- **Bounding box width and height** This is the width and length of the smallest rectangle that could contain the whole tuber.
- **Perimeter** This is the length of the outer edge of the tuber. Cracks and other deformities can potentially skew this number.
- **Convex perimeter** This is the length of the perimeter if the entire tuber shape followed a convex curve, as a circle or ellipse does. This can correct for increases in perimeter caused by cracks and deformities.
- **Area** This is the area contained in the perimeter. Cracks and other factors could subtract from this number
- **Convex hull area** This is the area contained with in the convex perimeter.
- **Compactness** This is a measure of how much the potato shape resembles a perfect circle based on the perimeter and area.
- **Roundness** This is a measure of how much the potato shape resembles a perfect circle based on the convex perimeter and convex hull area.
- **Maximum length** This is the greatest Euclidean distance between two points on the tuber. This is often a diagonal line across the tuber.

Skin Functions

TubAR measures three traits through the `find.skin()` function. Multiple photos can be processed as a batch by using `skin.all()`.

Function Inputs and Arguments

- `display =` : This determines whether R wil output an image with grayscale objects. This can be useful for ensuring all potatoes are being detected while debris is removed and for visualization of the analysis process. by default this is set to TRUE.
- `mode =` : If this is set to "debug" the display will show numbered objects and skinned area on the objects.
- `write.clean=` : If set to TRUE, an image of the potatoes with the background removed will be created, which may be wanted for situations such as presentations. This is set to FALSE by default.
- `pix min =` Sets the minimum pixel size of an object to not be removed with the background. This helps to remove dirt, flaked skin, or other debris that the function might otherwise attempt to measure as a potato. By default, this is set at 4000. If the function is measuring too many shapes per photo this can be increased (An order of magnitude increase is reasonable in trying to callibrate this). On the other hand, if small potatoes are being excluded the number can be lowered.
- `scaledown =` Sets the degree by which the image will be reduced in size to speed up processing. By default this is 4. Increasing the number will potentially speed up processing, however loss of detail will eventually lead to inaccurate measurements or errors.
- `colorcard =` Lists which corner the colorcard is positioned in. This is set to "bottomright" by default. "bottomleft", "topright", and "topleft" are the other settings. If color correction causes errors or otherwise is not desired, this can be set to F to skip this step. Skipping this step will likely decrease the accuracy of the skin data and is not recommended.

- `color.correct =` : This determines whether color correction will be preformed. This is set to TRUE by default
- `n.core =` : Determines the number of processor cores to use in the function. This is set to 1 by default. This setting is going to depend on your computer and is usually limited if running R through RStudio.

Trait Outputs

- **Skinning** This is an estimate of the percentage of skinned surface area of the tuber.
- **Redness** This is the average level of redness, as defined by the CIELAB colorspace, of the unskinned portion of the tuber.
- **Lightness** This is the average level of lightness, as defined by the CIELAB colorspace, of the unskinned portion of the tuber.

Exporting Data

The export functions `skin.export()` and `shape.export()` create a matrix of median values for each trait for each picture. this matrix is meant to be easy to turn into a csv or other file or to pass into other functions and programs. This function uses the name of the image file to identify clones in a picture so it is recommended that pictures be given names that can be easily tied back to a particular clone.

Tag Removal

If a tag is included in the photo, it can be removed at the data export stage. Tag removal is turned off by default to avoid unintended removal of true potatoes. `skin.export(data, remove_tag=T)` and `shape.export(data, remove_tag=T)` can be used to include tag removal in the exporting process. The way tag removal is accomplished in each fuction is different and affects how each function can be utilized. `export.skin(remove_tag=T)` removes data that is outside of a range of redness values. The thresholds of this range should be set at values that are inclusive of the redness values of any potatoes but where the tags would be excluded. Because redness is being used as the threshold, bright red or bright green tags work best to ensure these threshold values exist. By default, the green threshold is set at -5 and the red threshold is set at 50. These can be adjusted using the arguments `green_thresh` and `red_thresh`. `export.shape(remove_tag=T)` removes data from objects below a certain roundness threshold. This is generally a more wide ranging way to distinguish tags from potatoes and any square or rectangular tags should be excluded through this function. By default, the roundness threshold is set at 0.7. This can be adjusted using the argument `roundness_thresh`.

References

Further Reading

Caraza-Harter, M. V., & Endelman, J. B. (2020). Image-based phenotyping and genetic analysis of potato skin set and color. Crop Science, 60(1), 202-210.

Code Utilized

Donald Danforth Plant Science Center. (2020). Tutorial: Morphology Functions. Plantcv. https://plantcv.readthedocs.io/en/stable/morphology_tutorial/

Wollschlaeger, D., (2019). Convex hull, (minimum) bounding box, and minimum enclosing circle. http://dwoll.de/rexrepos/posts/diagBounding.html.

Package Imports

Bivand, R.S., Pebesma, E., Gomez-Rubio, V., 2013. Applied spatial data analysis with R, Second edition. Springer, NY. https://asdar-book.org/

Elzhov, T.V, Mullen, K.M, Spiess, A.N., and Bolker, B. (2016). minpack.lm: R Interface to the Levenberg-Marquardt Nonlinear Least-Squares Algorithm Found in MINPACK, Plus Support for Bounds. R package version 1.2-1, https://CRAN.R-project.org/package=minpack.lm

Pau, G., Fuchs, F., Sklyar, O., Boutros, M., and Huber, W, (2010): EBImage - an R package for image processing withapplications to cellular
phenotypes. Bioinformatics, 26(7), pp. 979-981, 10.1093/bioinformatics/btq046.

Pebesma, E.J., R.S. Bivand, 2005. Classes and methods for spatial data in R. R News 5 (2), https://cran.r-project.org/doc/Rnews/.

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Schlager S (2017). "Morpho and Rvcg - Shape Analysis in R." In Zheng G, Li S, Szekely G (eds.), *Statistical Shape and Deformation Analysis*, 217-256. Academic Press. ISBN 9780128104934.