Summarizing video effort and camera methods

Kara Leimberger

Overall goals:

1. Summarize video effort in various dimensions:

- Video hours (number of hours that cameras recorded)
- Number of cameras per replicate
- Number of plant species per replicate

2. Calculate additional summary stats and information for methods/supplemental methods:

- Percentage of cameras at stations ('station' = cluster of plants + cameras, 2 stations per site)
- Distance between stations
- What plant species were in pots (we brought these species in to create a 'floral array' at each station)

Step 1: Import filtered video data

Different data were used for different analyses, so effort will vary depending on the particular dataset. Here, I am just interested in two main datasets:

1. Data for experiment. This dataset was created in a previous script (01).
2. Data for 'normal' visitation. This is just the data from the 'pre' experimental period, so I will summarize data to level of experimental period (pre/post) and then just filter that summary.

```
# Data to analyze
data_experiment <- read.csv("../data/export/intermediate/Camera_data_filtered_for_analysis.csv")

# Function to calculate basic summary stats (mean, median, etc.)
source("../scripts/helper_functions/Calculate_basic_summary_stats.R")
```

Step 2: Calculate video effort (video hours) per replicate

Cameras were positioned at flowering plants and programmed to record for 12 hours/day (5:30-17:30). However, we do not have 12 hours/day of footage per camera for a variety of reasons:

- Sometimes cameras did not record for the entire time period due to technical difficulties (e.g., dead batteries or camera not switched from 'preview' mode to 'record' mode).
- People spent time around the focal plant, such as while completing routine tasks such as retrieving SD cards, replacing batteries, and watering potted plants. These activities would presumably dissuade hummingbirds from visiting the flowers on camera, so this video time was excluded during the video review process.

- Not all data were included in analysis. In a previous script (01), data were filtered to EXCLUDE videos from the (1) afternoon of covering day (in treatment replicates), (2) days with no flowers, (3) non-priority dates, and (4) cameras that didn't have video effort (or visible flowers) pre and post.

Here, I first to summarize to the camera level, rather than across plant species, to avoid inflating the video hours when >1 plant species in frame. When >1 plant species was present per camera, summarizing to species could lead to double/triple/quadruple counting of video hours.

```
# Summary at camera level
hours_per_camera_date <- data_experiment %>%
    distinct(year, patch, control_treatment, camera_num, date_video, exp_phase, file_id,
        video_length) %>%
    group_by(year, patch, control_treatment, camera_num, date_video, exp_phase) %>%
    summarise(video_hours = sum(video_length)) %>%
    ungroup()
```

Total effort

```
# Number of hours, summarized across cameras
hours_total <- hours_per_camera_date %>%
    summarise(video_hours = sum(video_hours)) %>%
    ungroup()

hours_total
```

```
## # A tibble: 1 x 1
##   video_hours
##         <dbl>
## 1       19870.
```

```
# How many total sightings occurred with confirmed visits? 'Sighting' =
# hummingbird appearing in the camera frame
videos_with_sightings <- data_experiment %>%
    filter(sightings_yes_or_no == "Y" & visit_type != "none")

# Number of sightings
(sightings_total <- unique(videos_with_sightings$sighting_id) %>%
    length())
```

```
## [1] 6475
```

```
# Note: number of sightings is less than number of rows in
# 'videos_with_sightings' because sometimes >1 plant in frame, and each plant
# gets a row for the same sighting.
```

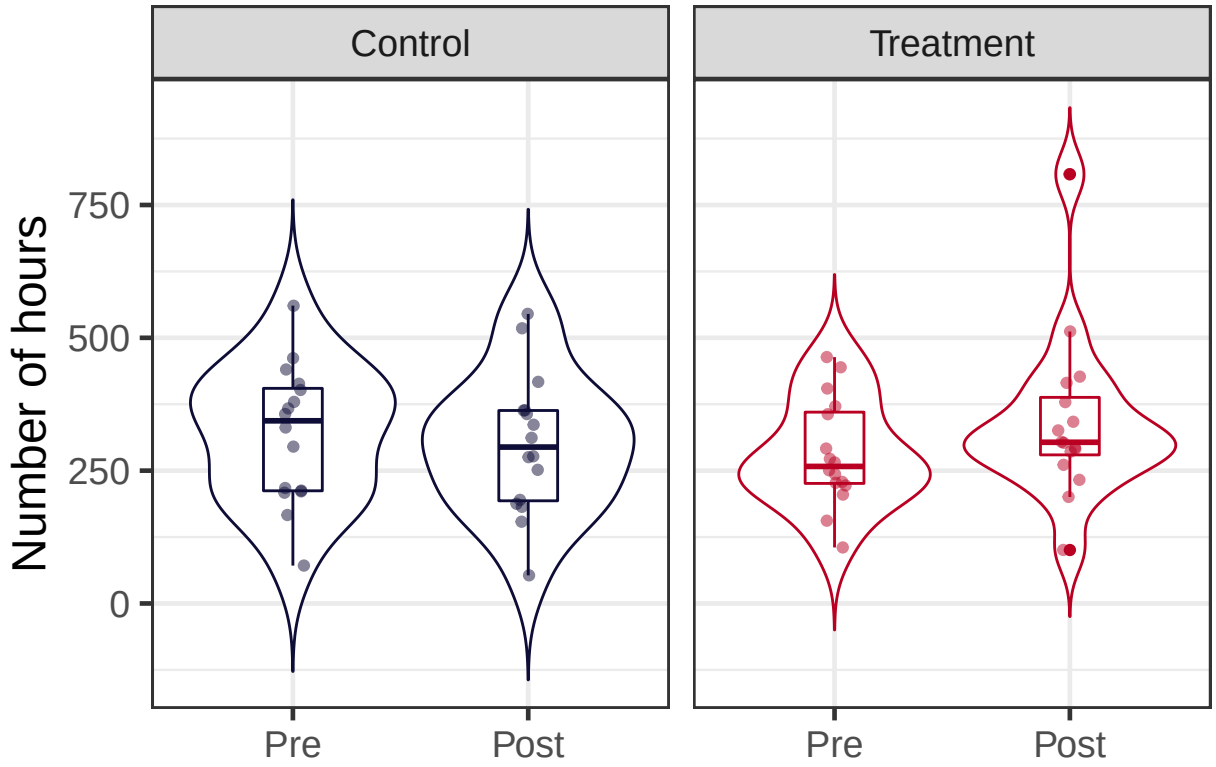
Effort for 'normal' visitation (pre period only)

```
# Number of hours, summarized across cameras
hours_total_normal_visitation <- hours_per_camera_date %>%
    filter(exp_phase == "pre") %>%
    summarise(video_hours = sum(video_hours)) %>%
    ungroup()

hours_total_normal_visitation
```

```
## # A tibble: 1 x 1
##   video_hours
##         <dbl>
## 1       9603.
```

Effort per experimental period

```
# pp = pre/post
hours_pp <- hours_per_camera_date %>%
    group_by(year, patch, control_treatment, exp_phase) %>%
    summarise(num_hours = sum(video_hours)) %>%
    ungroup()

# How much do video hours vary pre-to-post in control vs. treatment replicates?
hours_pp %>%
    mutate(control_treatment = factor(control_treatment, levels = c("control", "treatment"),
        labels = c("Control", "Treatment"))) %>%
    mutate(exp_phase = factor(exp_phase, levels = c("pre", "post"), labels = c("Pre",
        "Post"))) %>%
    ggplot(data = ., aes(x = exp_phase, y = num_hours, colour = control_treatment)) +
    geom_violin(position = position_dodge(width = 1), trim = FALSE) + geom_boxplot(position = position_
    width = 0.25) + facet_grid(. ~ control_treatment) + geom_point(position = position_jitterdodge(dodg
    jitter.width = 0.2, jitter.height = 0.2), alpha = 0.5) + labs(y = "Number of hours",
    x = "", colour = "") + scale_colour_manual(values = c("#0E0D37", "#BA0022"),
    guide = "none") + theme_bw(base_size = 18)
```



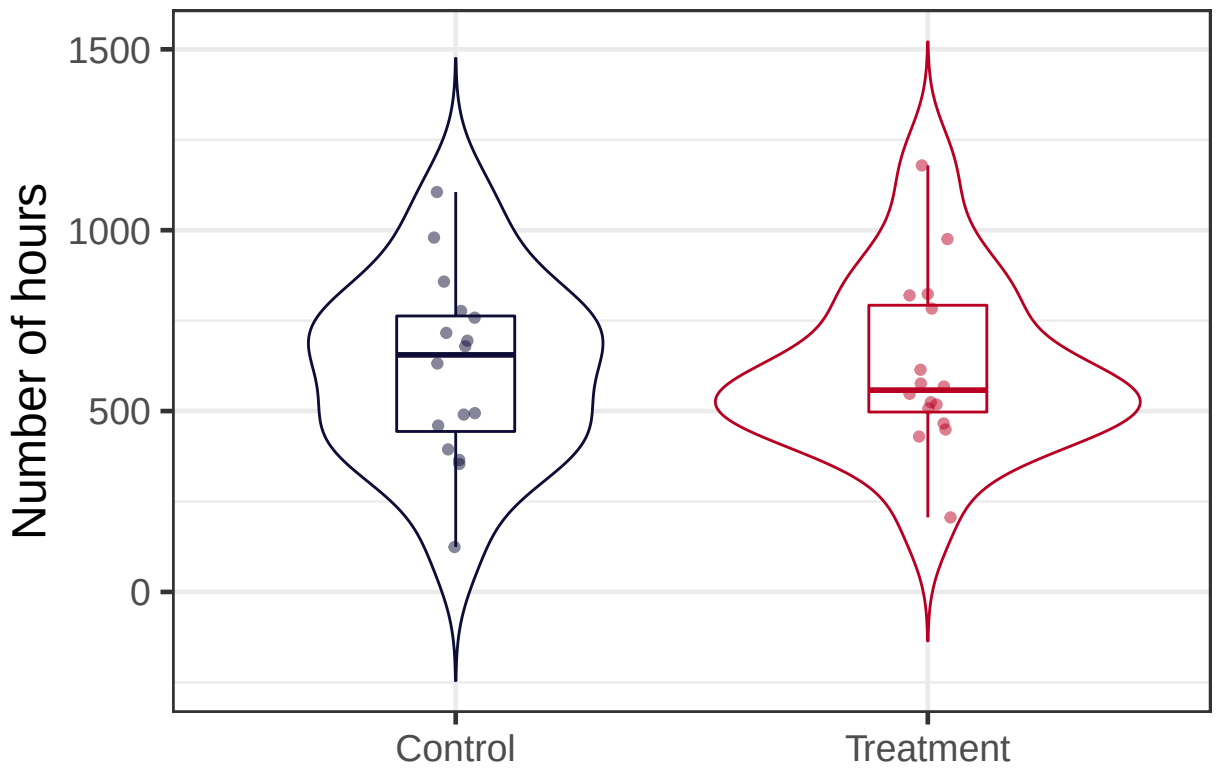
```r
# Summary of hours per experimental period
hours_pp_sum <- hours_pp %>%
    group_by(control_treatment) %>%
    calculate_basic_summary_stats(data = ., variable = num_hours)

hours_pp_sum
```

```
## # A tibble: 2 x 6
##   control_treatment   min   max  mean    sd median
##   <chr>             <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1 control            53.0  561.  309.  128.   322.
## 2 treatment         101.   808.  312.  133.   291.
```

Effort per replicate (control vs. treatment)

"Replicate" = each year + site combination. Due to switching treatments between years, a given site might be a control replicate or a treatment replicate (depending on the year)

```r
# Total hours per replicate (across experimental periods, i.e., pre and post
# combined)
hours_per_replicate <- hours_per_camera_date %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_hours = sum(video_hours)) %>%
    ungroup()

# How much do video hours vary in control vs. treatment replicates?
hours_per_replicate %>%
    mutate(control_treatment = factor(control_treatment, levels = c("control", "treatment"),
        labels = c("Control", "Treatment"))) %>%
    ggplot(data = ., aes(x = control_treatment, y = num_hours, colour = control_treatment)) +
    geom_violin(position = position_dodge(width = 1), trim = FALSE) + geom_boxplot(position = position_
    width = 0.25) + geom_point(position = position_jitterdodge(dodge.width = 1, jitter.width = 0.2,
    jitter.height = 0.2), alpha = 0.5) + labs(y = "Number of hours", x = "", colour = "") +
    scale_colour_manual(values = c("#0E0D37", "#BA0022"), guide = "none") + theme_bw(base_size = 18)
```




```
# Summary of hours/replicate
hours_per_replicate_sum <- hours_per_replicate %>%
    calculate_basic_summary_stats(data = ., variable = num_hours)

hours_per_replicate_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  124. 1179.  621.  243.   572.
```

Step 3: Calculate number of plant species + number of cameras per replicate

I can either calculate these numbers from:

1. The dataset of hummingbird sightings that will be used in analysis (i.e., the one I've been working with so far)

OR

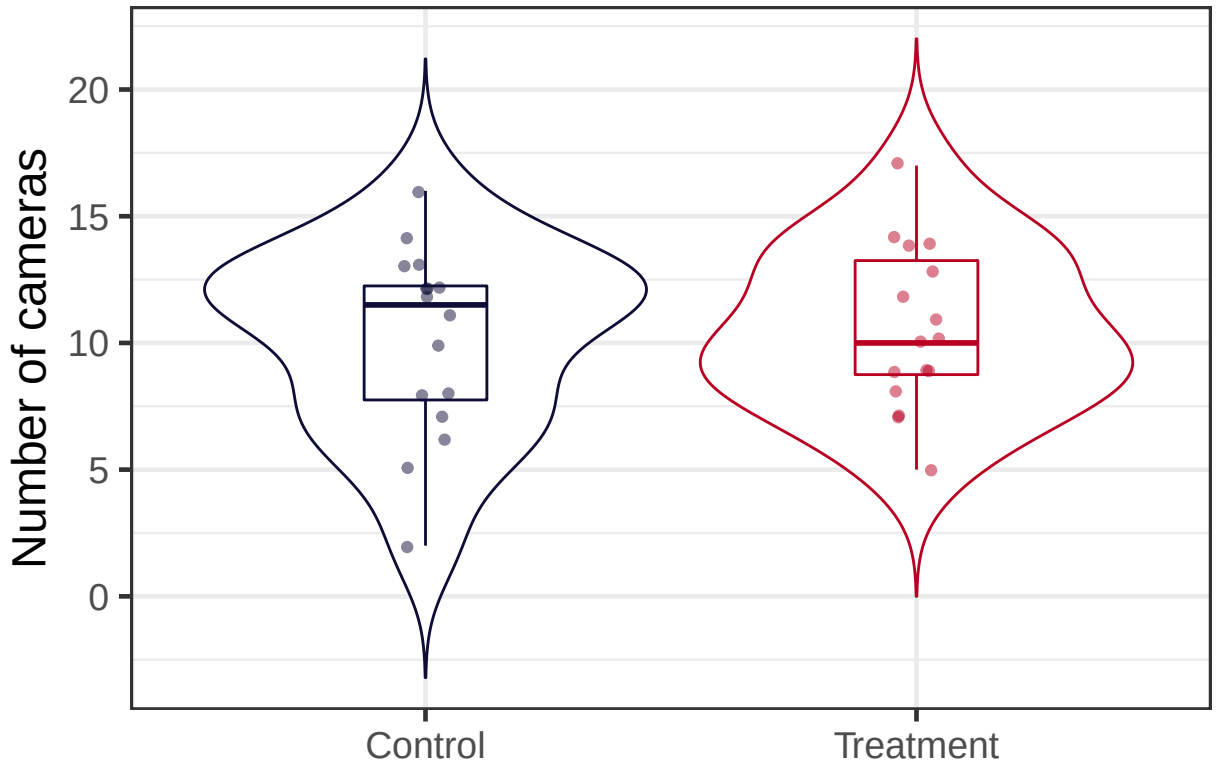
2. The dataset of camera locations (data recorded during camera installation)

The first option is more conservative, because not all cameras that were set up yielded usable data. For example, perhaps the focal plant had no flowers or data were unavailable for both experimental periods,

etc. However, in the methods section, it might be a bit more natural to say "We installed X cameras per replicate" rather than "We obtained usable data from X cameras per replicate".

Effort summary from sighting dataset (final, usable data)

```
# Number of cameras per replicate (there can be multiple plant species per
# camera)
cameras_per_replicate = data_experiment %>%
    distinct(year, patch, control_treatment, camera_num) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_cameras = n()) %>%
    ungroup() %>%
    filter(!is.na(patch))

cameras_per_replicate_sum = cameras_per_replicate %>%
    calculate_basic_summary_stats(data = ., variable = num_cameras)

# How much does the number of cameras/site vary between control and treatment
# replicates?
cameras_per_replicate %>%
    mutate(control_treatment = factor(control_treatment, levels = c("control", "treatment"),
        labels = c("Control", "Treatment"))) %>%
    ggplot(data = ., aes(x = control_treatment, y = num_cameras, colour = control_treatment)) +
    geom_violin(position = position_dodge(width = 1), trim = FALSE) + geom_boxplot(position = position_
    width = 0.25) + geom_point(position = position_jitterdodge(dodge.width = 1, jitter.width = 0.2,
    jitter.height = 0.2), alpha = 0.5) + labs(y = "Number of cameras", x = "", colour = "") +
    scale_colour_manual(values = c("#0E0D37", "#BA0022"), guide = "none") + theme_bw(base_size = 18)
```




```
# Number of plant species per replicate
species_per_replicate_sum = data_experiment %>%
    distinct(year, patch, control_treatment, plant_species) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_species = n()) %>%
    ungroup() %>%
    filter(!is.na(patch)) %>%
    calculate_basic_summary_stats(data = ., variable = num_species)

# Finally, just out of curiosity...which plant species are present most
# reliably across the different replicates?
replicates_per_species <- data_experiment %>%
    distinct(year, patch, control_treatment, plant_species) %>%
    group_by(plant_species) %>%
    summarise(num_replicates = n()) %>%
    ungroup() %>%
    arrange(desc(num_replicates))

head(replicates_per_species)
```

```
## # A tibble: 6 x 2
##   plant_species num_replicates
##   <chr>                  <int>
## 1 HETO                      31
## 2 MARA                      24
## 3 PAV-ROJ-AMA               22
```

```
## 4 CEPO                      21
## 5 TUBO-AMA                  18
## 6 HAPA                      17
```

Effort summary from camera setup dataset (record of cameras installed)

```
# Get camera location data
camera_locations = read.csv("../data/import/Camera_locations_2016-2018_20210705.csv")

# Number of cameras per replicate (there can be multiple plant species per
# camera)
cameras_per_replicate_setup = camera_locations %>%
    distinct(year, patch, control_treatment, camera_number_video_data) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_cameras = n()) %>%
    ungroup() %>%
    filter(!is.na(patch))

cameras_per_replicate_setup_sum <- cameras_per_replicate_setup %>%
    calculate_basic_summary_stats(data = ., variable = num_cameras)

# Number of plant species per replicate
species_per_replicate_setup_sum = camera_locations %>%
    distinct(year, patch, control_treatment, plant_species) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_species = n()) %>%
    ungroup() %>%
    filter(!is.na(patch)) %>%
    calculate_basic_summary_stats(data = ., variable = num_species)

# Number of plant species that were in pots (i.e., brought in to be part of
# focal array). This data was only recorded in 2017-2018
potted_plants_per_replicate_setup_sum <- camera_locations %>%
    filter(potted_y_n == "Y") %>%
    distinct(year, patch, control_treatment, camera_number_video_data) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_plants = n()) %>%
    ungroup() %>%
    calculate_basic_summary_stats(data = ., variable = num_plants)

potted_plants_per_replicate_setup_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <int> <int> <dbl> <dbl>  <dbl>
## 1     7    11  9.15  1.18      9
```

Compare methods

```
cameras_per_replicate_sum
```

```
## # A tibble: 1 x 5
```

```
##     min   max  mean    sd median
##   <int> <int> <dbl> <dbl>  <dbl>
## 1     2    17  10.3  3.44   10.5
```

```
cameras_per_replicate_setup_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <int> <int> <dbl> <dbl>  <dbl>
## 1     6    17  12.5  3.32     13
```

```
species_per_replicate_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <int> <int> <dbl> <dbl>  <dbl>
## 1     2    11  7.75  2.02      8
```

```
species_per_replicate_setup_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <int> <int> <dbl> <dbl>  <dbl>
## 1     6    14  9.28  2.05      9
```

As expected, more cameras were set up than yielded usable data. Results are similar for number of plant species as well.

Step 4: Calculate details of camera spatial arrangement

- Percent of cameras at stations
- Distance between stations

Calculate percentage of cameras at stations

```
# Note: these data might be a bit noisy, as I'm not sure how consistent field
# techs were in determining whether plant was 'at station' or not (i.e.,
# estimating how far from focal Heliconia plant it was and recorded it as being
# 'at station' or not)
cameras_per_replicate_station_setup = camera_locations %>%
    filter(station == "1" | station == "2") %>%
    distinct(year, patch, control_treatment, camera_number_video_data) %>%
    group_by(year, patch, control_treatment) %>%
    summarise(num_cameras_station = n()) %>%
    ungroup()

percent_cameras_at_station_setup_sum = cameras_per_replicate_setup %>%
    left_join(cameras_per_replicate_station_setup) %>%
    mutate(percent_at_station = num_cameras_station/num_cameras * 100) %>%
    calculate_basic_summary_stats(data = ., variable = percent_at_station)

percent_cameras_at_station_setup_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  33.3   100  83.7  17.1   85.8
```

On average, >80% of cameras at a station

Calculate distance between stations

Stations comprised a variety of different flowering plant species – some naturally occurring, some brought it in in pots – and was centered around a 'focal' Heliconia plant. There were two stations per site. How far apart were these stations?

Get data and write helper function for distance calculation

```
library(sf)
library(purrr)

# Projection to use
projstring <- "+proj=utm +zone=17 +ellps=WGS84 +units=m +no_defs"

# Function to calculate distance between locations of interest
calculate_distance <- function(data, group_id) {

    # Select data from same group
    data_same_group <- data %>%
        filter(replicate_id == group_id)

    # If there is >1 element, calculate distance between them
    if (length(data_same_group$replicate_id) > 1) {

        distance <- st_distance(data_same_group$geometry, by_element = FALSE)

        # First element in vector is zero; second element is actual distance
        return(distance[[2]])

    }

    return(NULL)

}

# Make a empty table to be filled in using purrr::map(). Replicate ID (patch +
# year combination) will be the group of interest (i.e., how far apart stations
# are within each replicate)
base_table <- camera_locations %>%
    distinct(year, patch) %>%
    mutate(replicate_id = paste(patch, year, sep = "_"))
```

Calculate distance using average station coordinates

```
# First, average coordinates assigned to each station (sometimes there are
# different coordinates, but presumably close to each other)
camera_stations <- camera_locations %>%
    distinct(year, patch, control_treatment, camera_number_video_data, plant_species,
```

```
        station, x, y) %>%
    filter(station == 1 | station == 2) %>%
    filter(!(is.na(x) | is.na(y))) %>%
    group_by(year, patch, station) %>%
    summarise(x = mean(x), y = mean(y)) %>%
    mutate(replicate_id = paste(patch, year, sep = "_"))

camera_stations_spatial <- st_as_sf(camera_stations, coords = c("x", "y"))
camera_stations_spatial <- st_set_crs(camera_stations_spatial, value = projstring)

station_distance = base_table %>%
    mutate(distance = map(replicate_id, ~calculate_distance(data = camera_stations_spatial,
        group_id = .))) %>%
    unnest(distance) %>%
    arrange(replicate_id)

# 1. Remove 'unit' attributes (meters) 2. Summarize
station_distance_sum <- station_distance %>%
    mutate(distance = units::drop_units(distance)) %>%
    calculate_basic_summary_stats(data = ., variable = distance)
```

Calculate distance using distance of focal Heliconia locations

Stations should be centered around focal Heliconia, so results should be similar as station results

```
focal_heto_locations <- camera_locations %>%
  distinct(year, patch, control_treatment, camera_number_video_data, plant_species, station, x, y) %>%
  filter((station == 1 | station == 2)) %>% #Note: by only looking at station numbers, this omits 2017 p
  filter(plant_species == "HETO") %>%
  filter(!(is.na(x) | is.na(y))) %>%
  group_by(year, patch, station) %>%
  summarise(x = mean(x),
            y = mean(y)) %>%
  mutate(replicate_id = paste(patch, year, sep = "_"))

focal_heto_locations_spatial <- st_as_sf(focal_heto_locations, coords = c("x", "y"))
focal_heto_locations_spatial <- st_set_crs(focal_heto_locations_spatial, value = projstring)

station_distance_focal_heto = base_table %>%
  mutate(distance = map(replicate_id, ~calculate_distance(data = focal_heto_locations_spatial, group_id
  unnest(distance) %>%
  arrange(replicate_id)

#1. Remove 'unit' attributes (meters) 2. Summarize
station_distance_focal_heto_sum <- station_distance_focal_heto %>%
  mutate(distance = units::drop_units(distance)) %>%
  calculate_basic_summary_stats(data = ., variable = distance)
```

Compare methods

```
station_distance_sum
```

```
## # A tibble: 1 x 5
```

```
##     min   max  mean    sd median
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  21.2  229.  59.4  44.4   48.0
```

```
station_distance_focal_heto_sum
```

```
## # A tibble: 1 x 5
##     min   max  mean    sd median
##   <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1  24.0  236.  61.6  43.7   55.6
```

Results are indeed similar if use station locations vs. focal Heliconia. Median of ~50m, mean of ~60 m +/- 44 m SD

Step 5: Summarize which plant species were present in focal arrays at stations (for table in Supplemental Methods)

```
# Note: potted status was only recorded in 2017-2018, but plants used in these
# years encompass all of 2016 species except IMPO (which did not yield any
# usable camera data anyway)
floral_array_species <- camera_locations %>%
    filter(potted_y_n == "Y") %>%
    distinct(plant_species)

floral_array_species
```

```
##    plant_species
## 1       TUBO-AMA
## 2           AMAP
## 3          GOLFO
## 4       PAV-ROSA
## 5    PAV-ROJ-AMA
## 6           COWO
## 7           HETO
## 8           PALU
## 9           RABO
## 10        SASPLE
## 11          HAPA
## 12     BOCA-ROJA
## 13          CEPO
## 14    RABO, PALU
## 15       COSTBAR
```

```
# Add scientific name, etc. for supplement
plant_info <- read.csv("../data/import/Plant_codes_species_list.csv") %>%
    select(family, species_name, plant_species = species_code)

floral_array_species_supp <- floral_array_species %>%
    left_join(plant_info) %>%
    filter(!is.na(family)) %>%
```

```
    arrange(family, species_name) %>%
    filter(plant_species != "HETO")  #Remove HETO. Not really part of floral array/potted plants

# Where there any species recorded in setup data that didn't end up in usable
# data?
check01 <- floral_array_species_supp %>%
    filter(!(plant_species %in% data_experiment$plant_species))

# Export
write.csv(floral_array_species_supp, "../data/export/supp_tables/List_of_floral_array_species_for_Supple
```