Version control

why you should want it

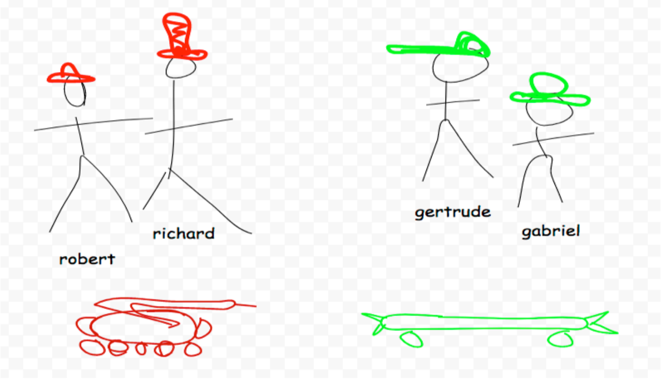
Version control and intro to git

Learning objectives

- Desire to use version control for *everything*
- Not be scared of git
- Some practise

An inspirational story

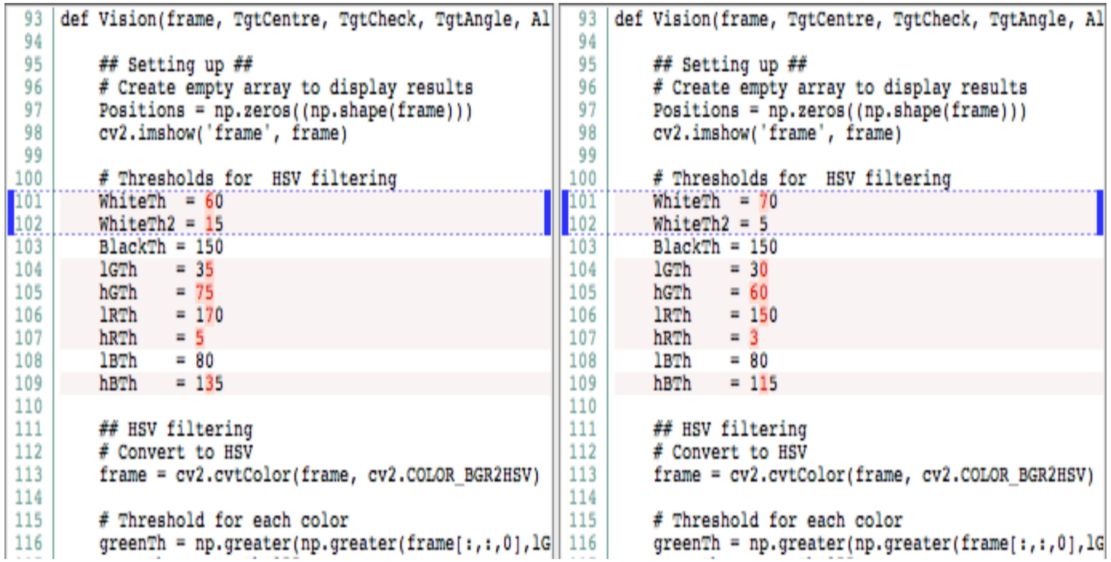
(based in a parallel universe where git does not exist)



MONDAY

TUESDAY

WEDNESDAY



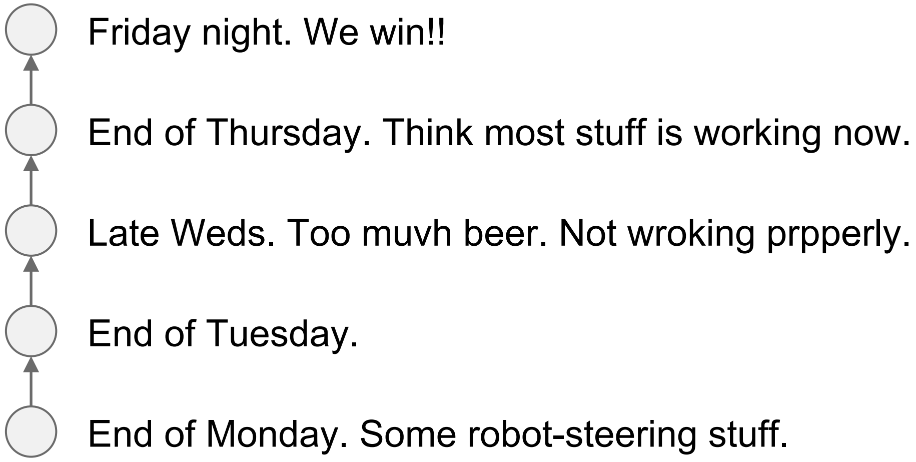
```
def Vision(frame, TgtCentre, TgtCheck, TgtAngle, Al

    ## Setting up ##
    # Create empty array to display results
    Positions = np.zeros((np.shape(frame)))
    cv2.imshow('frame', frame)

    # Thresholds for  HSV filtering
    WhiteTh  = 60
    WhiteTh2 = 15
    BlackTh = 150
    lGTh    = 35
    hGTh    = 75
    lRTh    = 170
    hRTh    = 5
    lBTh    = 80
    hBTh    = 135

    ## HSV filtering
    # Convert to HSV
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Threshold for each color
    greenTh = np.greater(np.greater(frame[:,:,0],lG
```

```
def Vision(frame, TgtCentre, TgtCheck, TgtAngle, Al

    ## Setting up ##
    # Create empty array to display results
    Positions = np.zeros((np.shape(frame)))
    cv2.imshow('frame', frame)

    # Thresholds for  HSV filtering
    WhiteTh  = 70
    WhiteTh2 = 5
    BlackTh = 150
    lGTh    = 30
    hGTh    = 60
    lRTh    = 150
    hRTh    = 3
    lBTh    = 80
    hBTh    = 115

    ## HSV filtering
    # Convert to HSV
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

    # Threshold for each color
    greenTh = np.greater(np.greater(frame[:,:,0],lG
```

THURSDAY

FRIDAY
Competition day!

A simple workflow



Friday night. We win!!

End of Thursday. Think most stuff is working now.

Late Weds. Too muvh beer. Not wroking prpperly.

End of Tuesday.

End of Monday. Some robot-steering stuff.

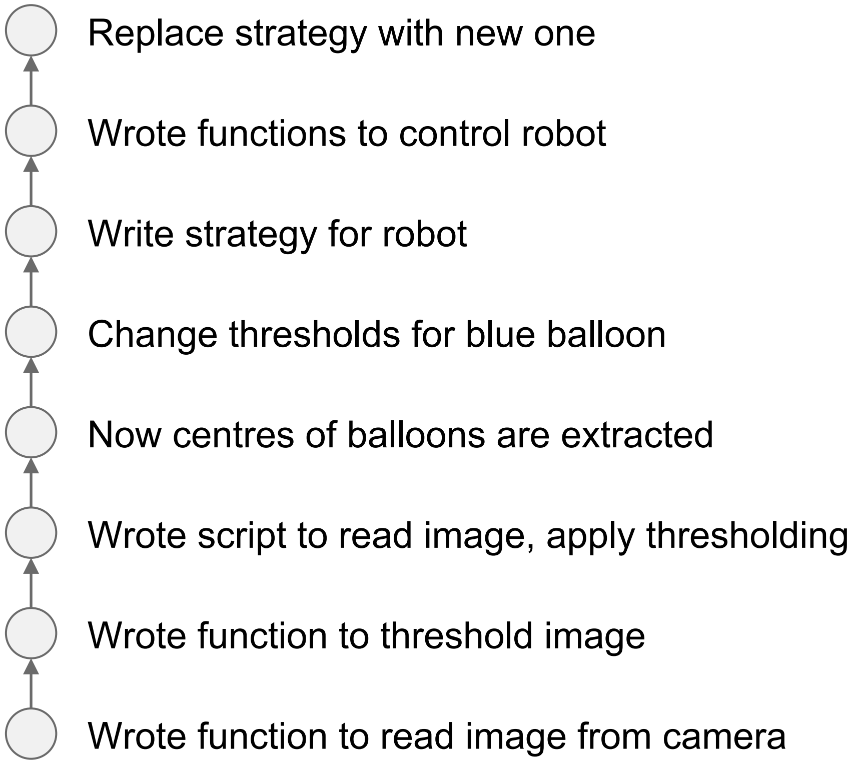
What do we win?

- Saving points
    - What did the code look like yesterday? (I think it was working then…)
    - Adam wants to see our light-following robot, but we already changed it to a balloon-popping strategy. Let’s switch to the old version for a demo.
- Narrow down bugs
    - The robot works in v1 and v2, so the bug must be introduced in v3
- See what changed
    - Aha, Steve ~~broke~~ changed the tuning for the light-following!
- Pin down “special” versions
    - This is the version of the program we were running when the AI gained consciousness and held our team-mate hostage - let’s interrogate the code to figure out how to stop it

But remember… Code is Text

All your programming is text. Everything is ascii, changes are (ideally) clustered.

Programming is **Lego**



Replace strategy with new one

Wrote functions to control robot

Write strategy for robot

Change thresholds for blue balloon

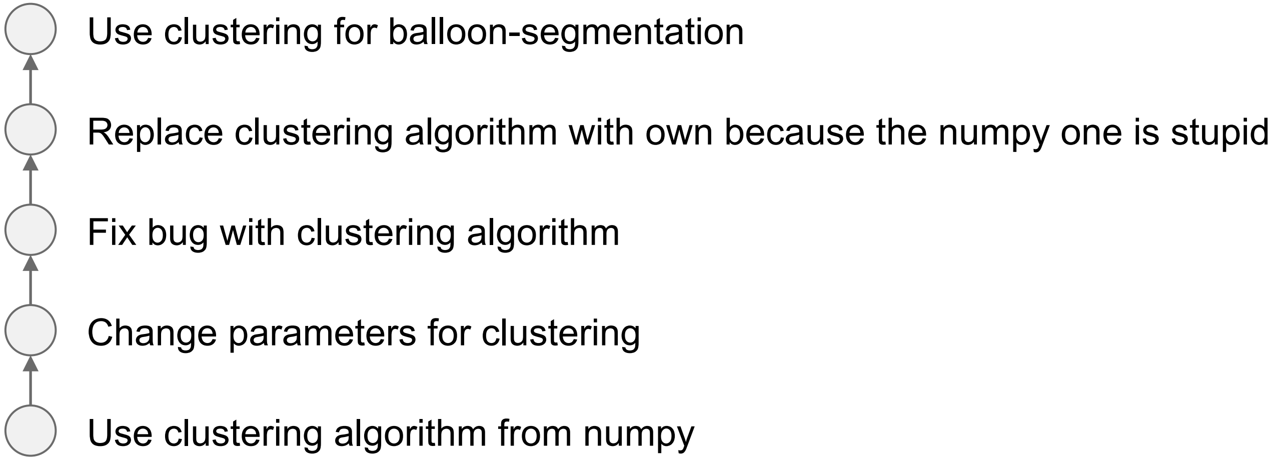
Now centres of balloons are extracted

Wrote script to read image, apply thresholding

Wrote function to threshold image

Wrote function to read image from camera

Programming is **a story**



Use clustering for balloon-segmentation

Replace clustering algorithm with own because the numpy one is stupid

Fix bug with clustering algorithm

Change parameters for clustering

Use clustering algorithm from numpy

Really useful things you can do with version control

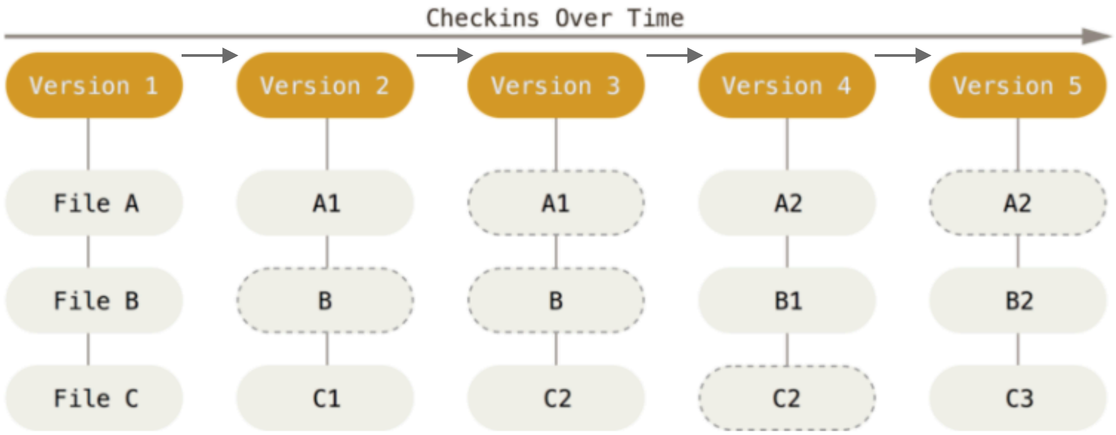
- Better debugging
  - What did I change in the last 20 minutes of late-night debugging to make it work?
  - Which of these changes were necessary? Which were superfluous?
- What was the change that broke X?
  - Who is responsible?
- Which version of the code did I use for these results?
  - All data/analysis used for publication should be reproducible
  - Can you guarantee this a year after submitting the paper?
- Easy switching between work
  - Quickly re-run the analysis your supervisor asked for whilst in the middle of new coding

Really useful things you can do with version control

- Visibility of others’ work
    - What has my collaborator added since our last meeting?
    - Who wrote this terrible line of code?
- Split work up
    - I’ll work on this part, you work on that, we’ll combine our changes when we’re ready
- Easy switching between work
    - Try out your collaborator’s new code without losing your changes
    - Get a bugfix from your collaborator’s work without taking their still-buggy additions
    - Share some of the things you’ve changed without sharing the still-buggy stuff
    - Publish a stable version publicly, work on tentative new stuff privately

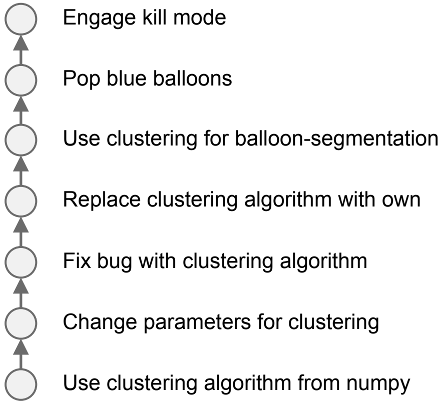
How git works

- Git provides a database handling version control for us
  - Stores a zip file with each version of each file
  - .git directory holds all the magic



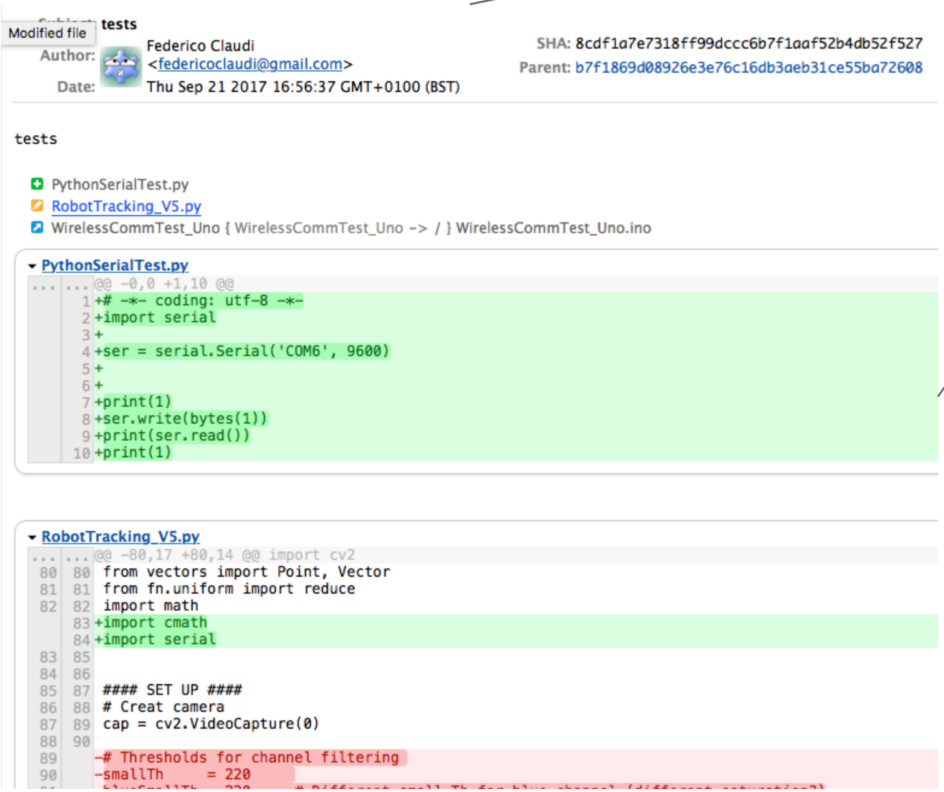
How to use git

- We need to tell git what to put in each zip file
  - Which files should it “track” ?
  - What changes are relevant to this “commit”
  - Choose what to “stage” (add to a “commit”)
    - Which new files?
    - Which changes to tracked files? (per line)
    - Be aware of .gitignore for files you never want to track
    - (switch to gitx demo)
- Once it’s in git, it is safe (you can only *add* to a git repo)
  - You can “check out” any of your historical versions, and your local files will be updated to reflect that version.
- Think of commits as representing the *changes* to the code


Engage kill mode
Pop blue balloons
Use clustering for balloon-segmentation
Replace clustering algorithm with own
Fix bug with clustering algorithm
Change parameters for clustering
Use clustering algorithm from numpy

Commits as diffs

- New files
- Changes to files
- Deleted files



Modified file tests

Author: Federico Claudi <federicoclaudi@gmail.com>

Date: Thu Sep 21 2017 16:56:37 GMT+0100 (BST)

SHA: 8cdf1a7e7318ff99dccc6b7f1aaf52b4db52f527

Parent: b7f1869d08926e3e76c16db3aeb31ce55ba72608

tests

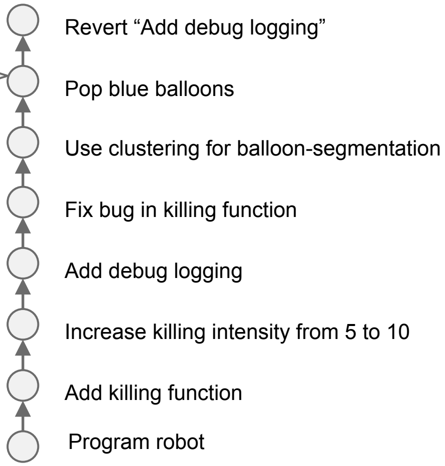
- PythonSerialTest.py
- RobotTracking_V5.py
- WirelessCommTest_Uno { WirelessCommTest_Uno -> / } WirelessCommTest_Uno.ino

PythonSerialTest.py

```
@@ -0,0 +1,10 @@
+# -*- coding: utf-8 -*-
+import serial
+
+ser = serial.Serial('COM6', 9600)
+
+
+print(1)
+ser.write(bytes(1))
+print(ser.read())
+print(1)
```

RobotTracking_V5.py

```
@@ -80,17 +80,14 @@ import cv2
 from vectors import Point, Vector
 from fn.uniform import reduce
 import math
+import cmath
+import serial


 #### SET UP ####
 # Creat camera
 cap = cv2.VideoCapture(0)

-# Thresholds for channel filtering
-smallTh      = 220
-blueSmallTh  = 220       # Different smallTh for blue channel (different saturation?)
```



Revert “Add debug logging”

Pop blue balloons

Use clustering for balloon-segmentation

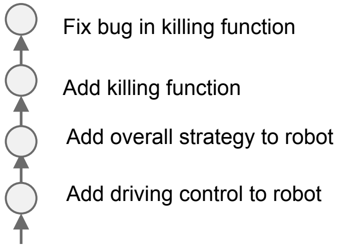
Fix bug in killing function

Add debug logging

Increase killing intensity from 5 to 10

Add killing function

Program robot



Fix bug in killing function

Add killing function

Add overall strategy to robot

Add driving control to robot

But what about github?

- Github is a website that allows you to save git repositories publicly or privately
- (also Bitbucket , Gitlab, other options available)

This allows you to:

- Back up your repositories
- Share your work publicly
- Collaborate with other people

All of these things can be done without github.

Conflicts!

- What if two collaborators make non-compatible changes?
    - Git is pretty smart at resolving most conflicts
    - Sometimes it just can’t know what the “correct” answer is unless you tell it.
- Types of changes:
    - Kirsty added some text in her branch
    - Steve removed some text in his branch
    - Kirsty and Steve both modified the same text in their respective branches
- Exercise:
    - See handout