ROOT SEGMENTATION AND ROBOTIC CONTROL FOR PLANT-MICROBE INTERACTIONS

THIJN BAKKER

LAYOUT

INTRODUCTION

- Who is the client?
- What do they want?
- What is the Project Context?

PROJECT OBJECTIVES

- Segment plant roots from images.
- Accurate segmentation models.

- Control a robotic system to inoculate plants precisely.
- Robotic control system integrated with computer vision.

METHODOLOGY OVERVIEW

PHASE 1: IMAGE PROCESSING AND SEGMENTATION

PHASE 2: DATA ANALYSIS

PHASE 3: ROBOTICS INTEGRATION

KEY CHALLENGES

I. Complexities in root segmentation

II. Dataset bias and limited root masking

III. Precision requirements for robotic inoculation

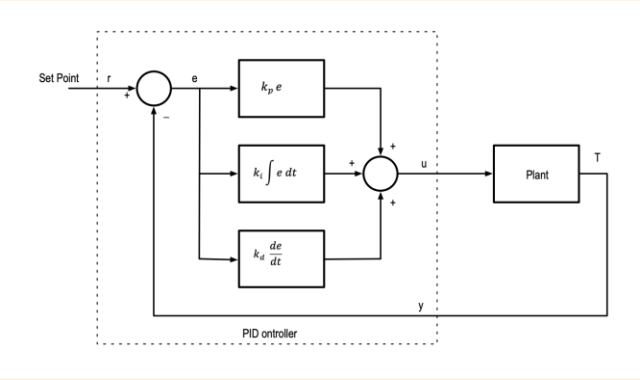


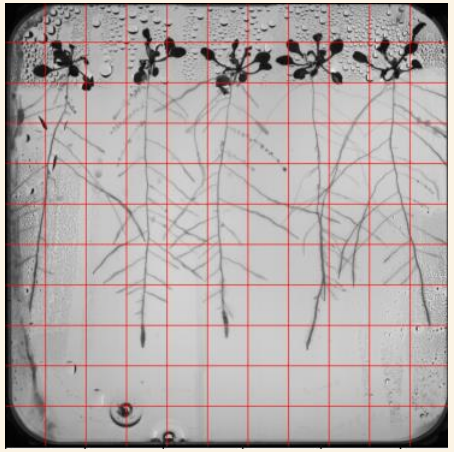
IMAGE SEGMENTATION PIPELINE

- Original image (1)
- Cropped image (2)
- Padded image (3)
- Patched image (4)

1. 

2. 

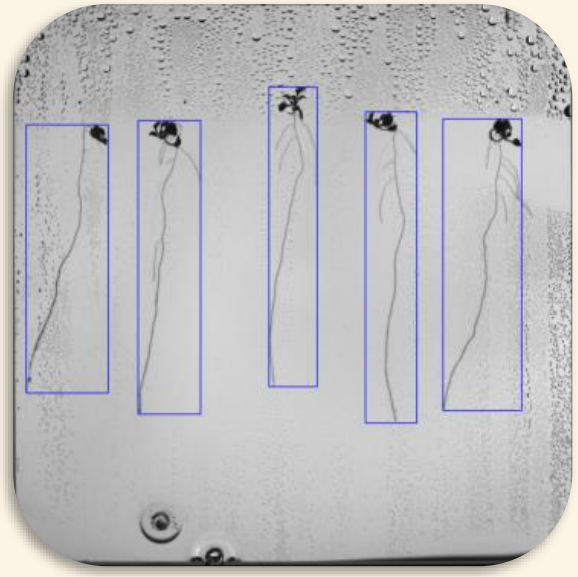
3. 

4. 

MODEL TRAINING (1)

First Iteration:

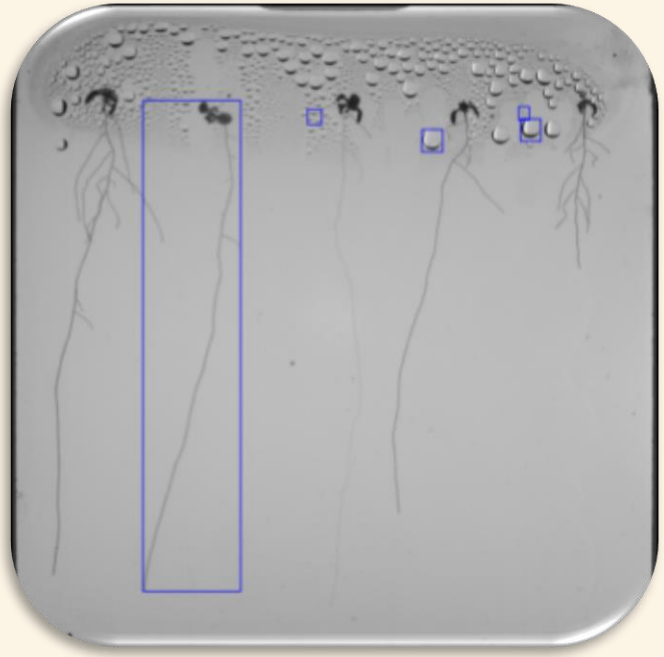
- No model was used
- Bounding boxes around the roots
- Did NOT devide roots into regions
- Applied connected components
- Top 5 stats (roots) in the image

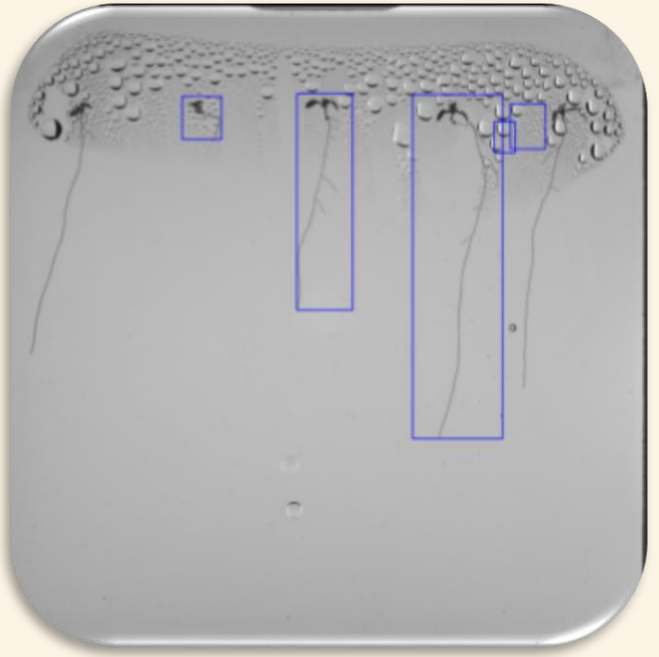


- Kaggle result:
  - Public score = 110.174
  - Private score = 90.384

MODEL TRAINING (2)

- Kaggle result:
  - Public score = 110.174
  - Private score = 90.384





MODEL TRAINING (3)

Best Iteration:

- A Simple U-Net model was used
- While monitoring the val_f1
- Best validation f1: 0.857

- Hyperparameters:
  - Batch size = 32
  - Epochs = 100
  - Patch size = 256
  - Step size = 128
  - Learning rate = 1e-3

- Callbacks:
  - ReduceLROnPlateau
  - EarlyStopping
  - WandCallBack

EXTRACTING VALUES (1)

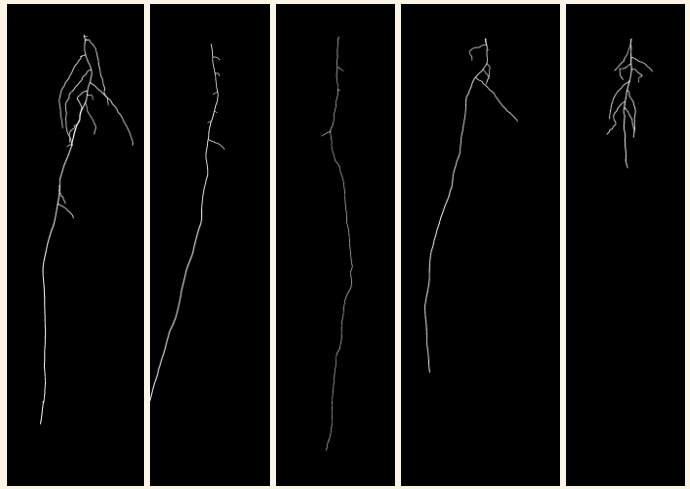
Step 1: Separate the root masks into regions

Step 2: Skeletonize the roots

Step 3: Get the branch data

Step 4: Detect the top and bottom nodes

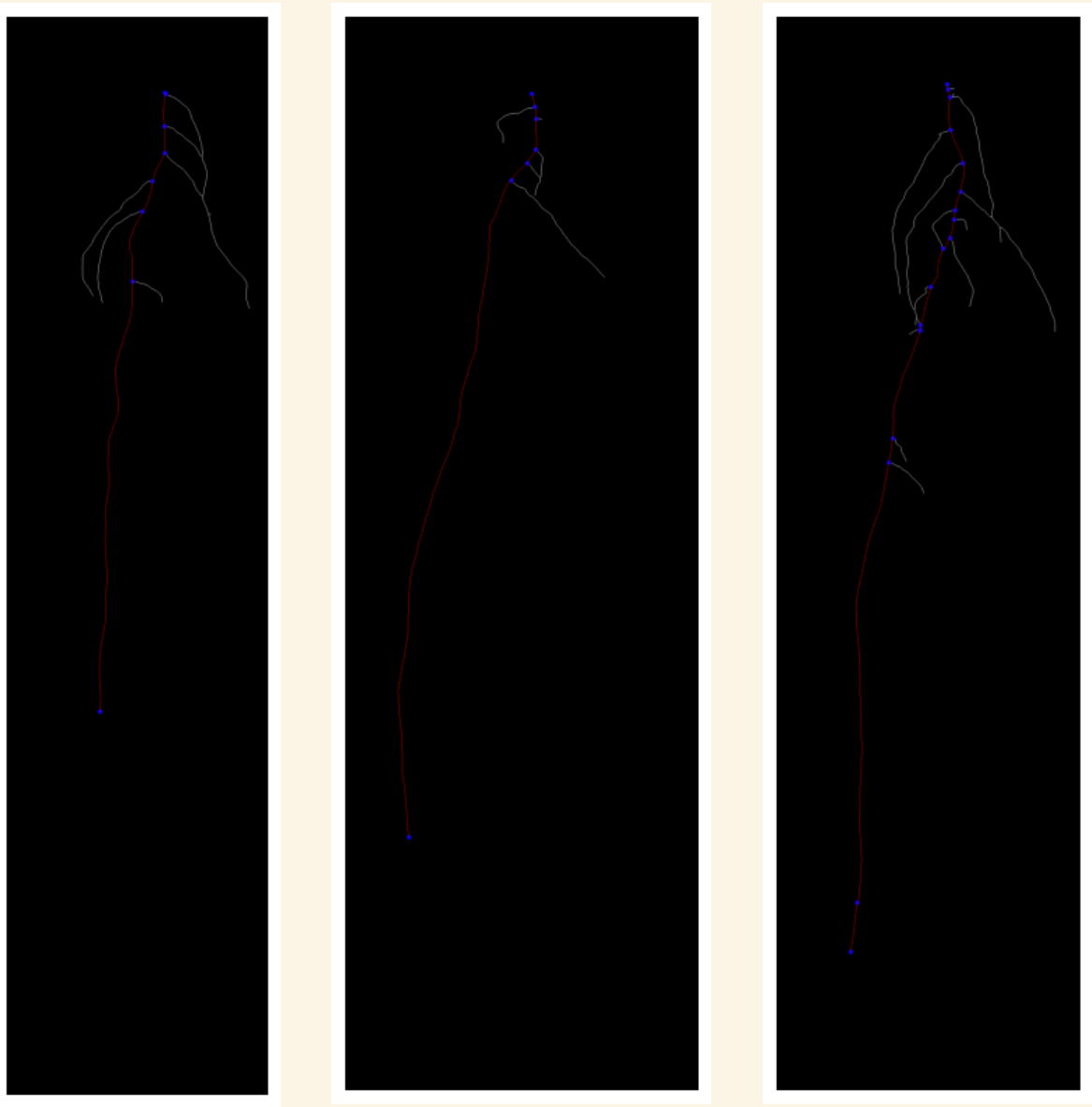
Step 5: Calculate the length of the root



	skeleton-id	node-id-src	node-id-dst	branch-distance
0	0	0	5	5.414214
1	0	5	862	457.457936
2	0	5	247	93.142136
3	0	247	265	10.071068
4	0	247	432	90.313708
5	0	432	852	198.977705
6	0	432	908	189.455844
7	0	908	1280	197.066017
8	0	908	1176	127.242641
9	0	1176	1410	111.539105
10	0	1176	1243	13.414214

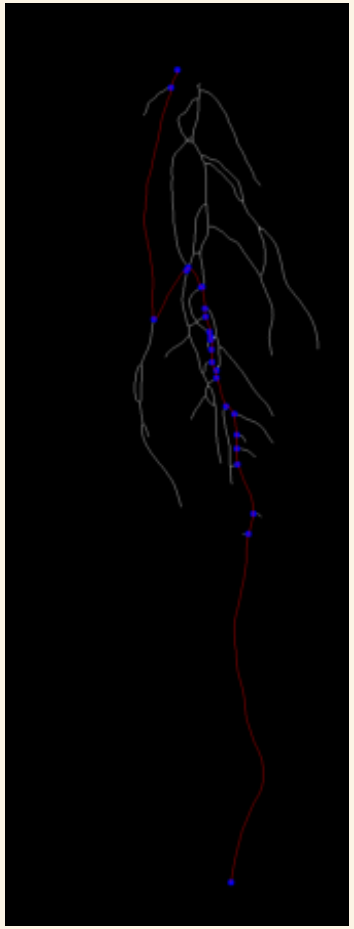
- Regions: [(10, 700), (600, 1200), (1100, 1700), (1500, 2300), (2100, 2700)]

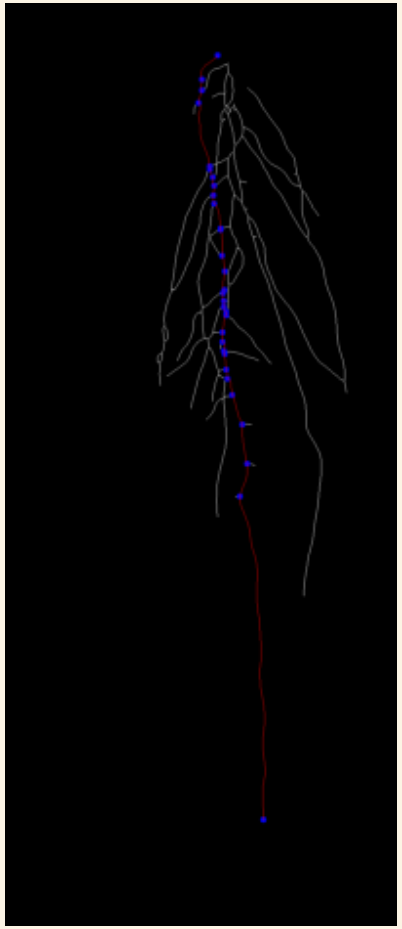
EXTRACTING VALUES (2)



- Kaggle result:
  - Public score = 2.445
  - Private score = 6.807

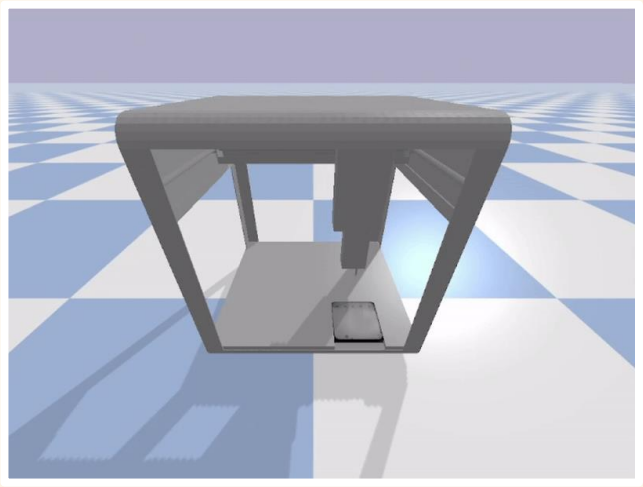
EXTRACTING VALUES (3)

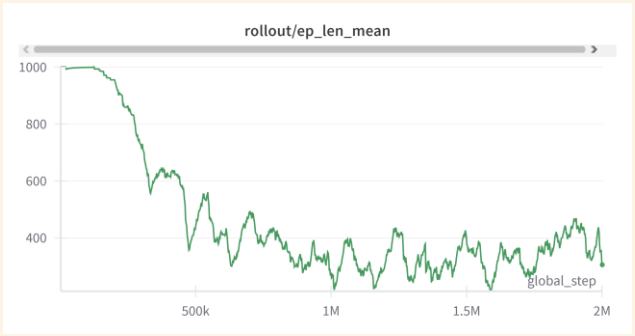


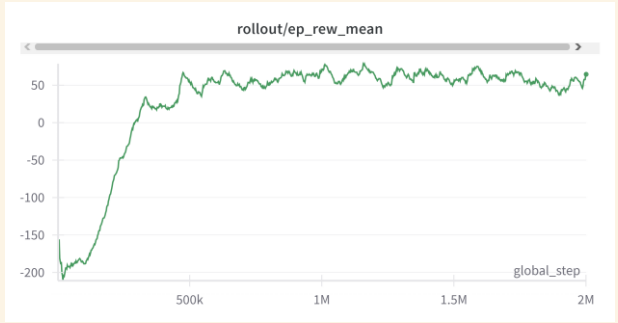


ROBOTIC CONTROL SYSTEM(1)

- Set up a simulation environment for the Opentrons OT-2
- Trained RL-based controllers for task optimization.
- Designed a PID controller for precision







reward = - np.linalg.norm(pipette_pos - goal_pos) If it reaches the goal: reward += 100.0

ROBOTIC CONTROL SYSTEM (2)

First Iteration:

- Connected the gym wrapper to the PID controller
- Forgot the working envelope
- Used a different goal position each time
- Did not reach the goal in 1000 steps

Start Position	[0.073, 0.0895, 0.1195]
Goal position	[3.0969, 8.1294, -1.2896]
Control Signal	[1.000e-04, 2.000e-04, 1.195e-01]
Final Position	[1.503e-08, 1.418e-08, 1.195e-06]

- PID values:
  - Kp = 100.0
  - Ki = 0.5
  - Kd = 0.01

ROBOTIC CONTROL SYSTEM (3)

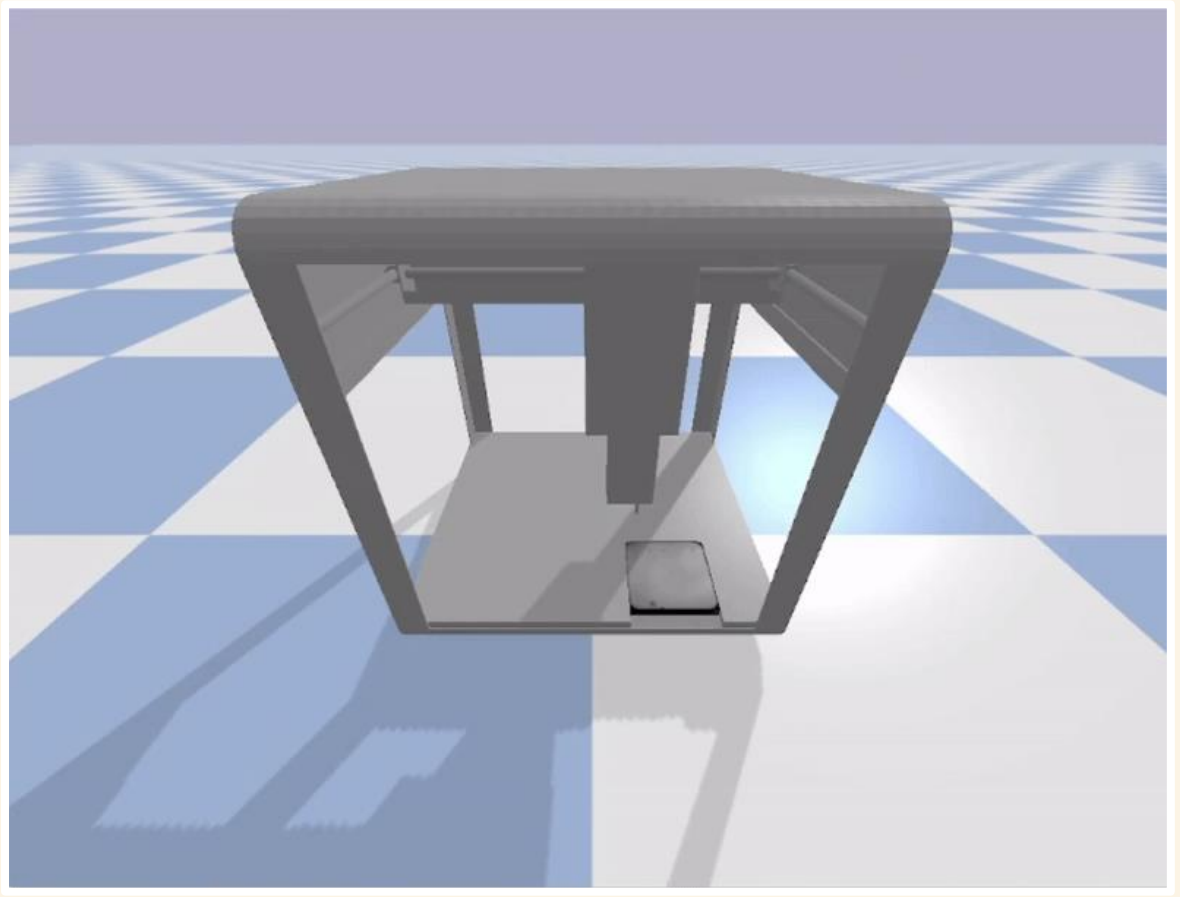
Best Iteration:

- Connected the PID controller to the Simulation (sim_class)
- Used the correct working envelope
- Used a different startin/goal position each time
- Reached goal in 192 steps

Start Position	[0.1077, 0.0880, 0.057]
Goal position	[0.0758, -0.0705, 0.2510
Control Signal	(-0.1684, -0.1770, 0.1044)
Final Position	[-0.0792, -0.0707, 0.2498]

- PID values:
  - Kp = 1.0
  - Ki = 0.01
  - Kd = 0.1
- Working Envelope:
  - x = [-0.1875, 0,253]
  - y = [-0.1705, 0,2197]
  - z = [0.1197, 0,2896]

ROBOTIC CONTROL SYSTEM (4)



FUTURE RECOMMENDATIONS

- CONNCECTING CV TO RL&PID
- COMPARE RL TO PID
- DATA AUGMENTATION
- FIX MAIN ROOT DETECTION ERRORS

CONCLUSION

THANK YOU

THIJN BAKKER

231131@BUAS.NL

WWW.DONEWITHTHESEROOTS.COM