# BotROS: The Joy of Painting Progress Update

Members:
Sam Williams (will6673), Nadya Postolaki (posto018), Alex Overman (ayres036)

# What are we doing?

- Give BotROS (TurtleBot with a Bob Ross wig) an image and he “draws” it on a plane.
- Using the burger model

# Why are we doing this?

---

“I think there’s an artist hidden at the bottom of every single one of us.” - Bob Ross

# Why is it relevant to CSCI 5551?

- ROS and RViz
- Inverse Kinematics
- Quaternions for orientation

# What we've done so far and who's doing what

Nadya

- Studying the vectorization of images.
- Finding Python 3 solutions to creating vector images to provide TurtleBot.

Alex

- ROS & TurtleBot 3 installation
- Programming the path-finding aspects of BotROS in Python3

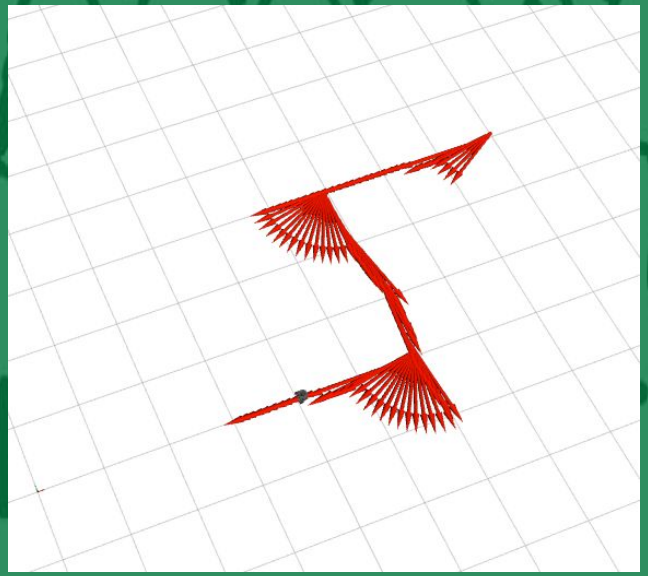
Sam

- ROS & TurtleBot 3 installation (partial)
- Adding Drawing Functionality (Pen) to TurtleBot 3

# What do your results look like?

---

~~Not good~~ BotROS is trying his best.



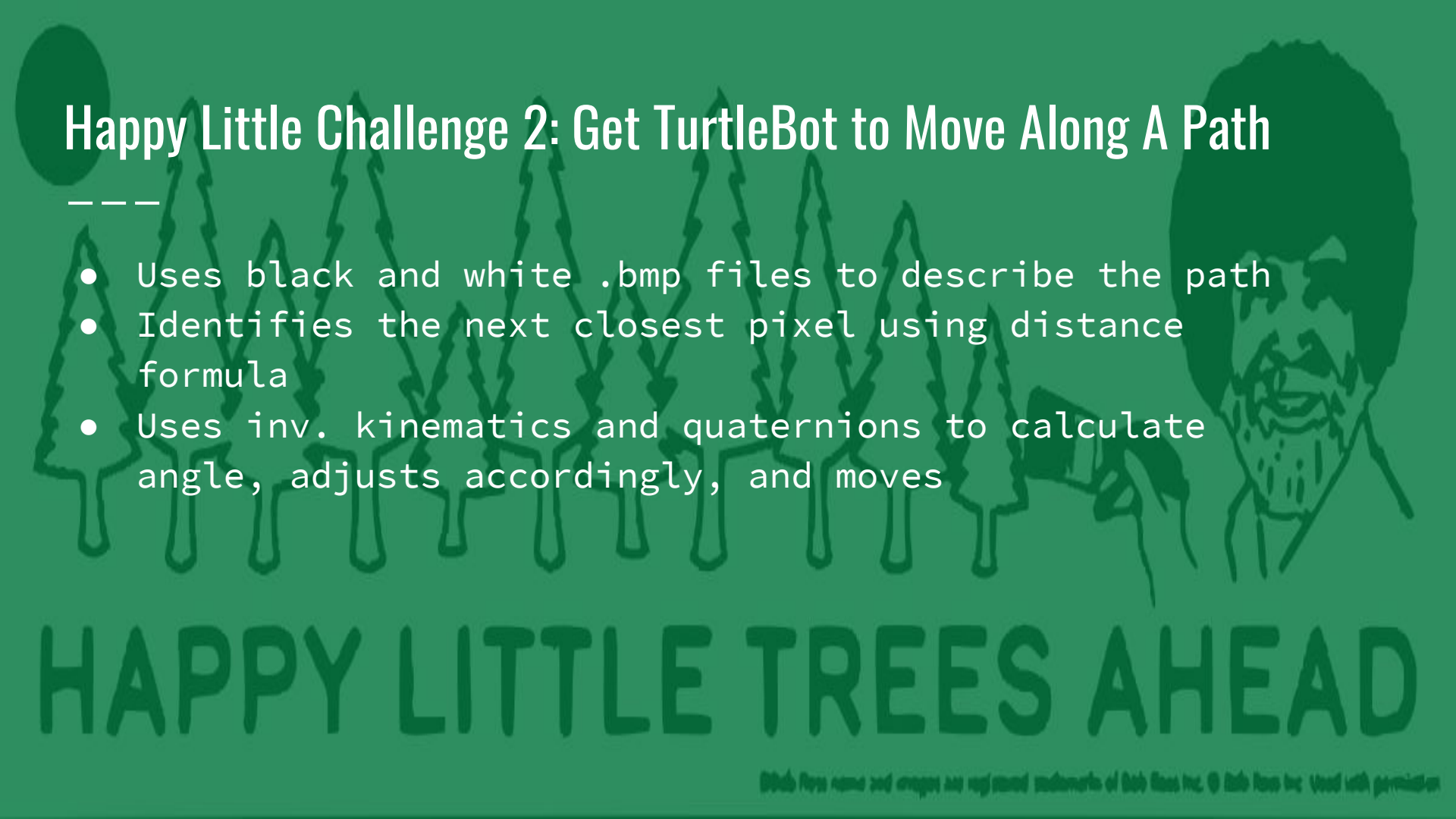
# Happy Little Challenge 1: Get TurtleBot to Move

- clone TurtleBot Package repos
- Generate TurtleBot environment
- Operate TurtleBot with controls or code

# Happy Little Challenge 2: Get TurtleBot to Move Along A Path

---

- Uses black and white .bmp files to describe the path
- Identifies the next closest pixel using distance formula
- Uses inv. kinematics and quaternions to calculate angle, adjusts accordingly, and moves

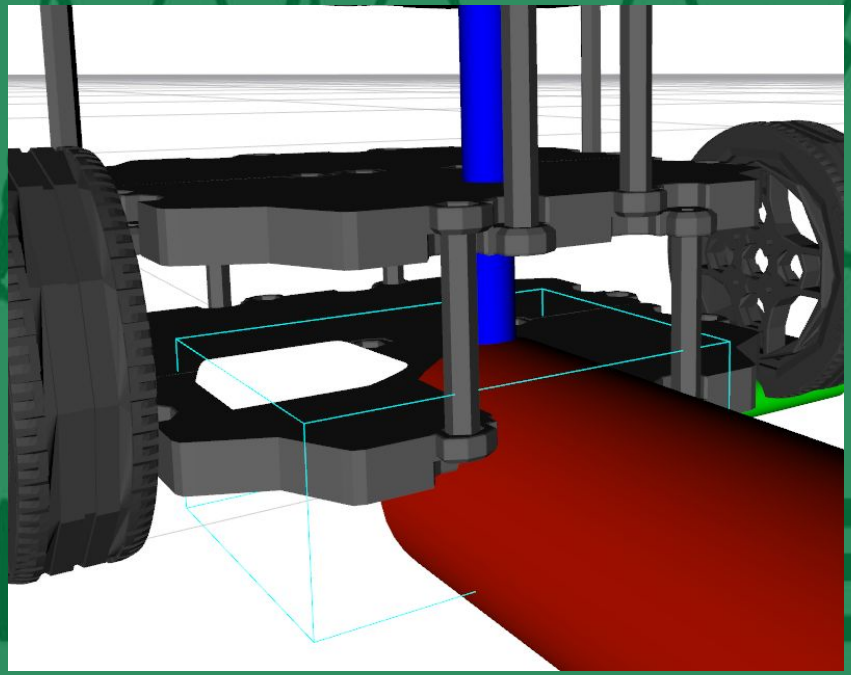


# Happy Little Challenge 2 (continued)

---

- Subscribes to the odometry topic to update the current position and orientation
- Publishes to the cmd_vel topic to change the position

# Happy Little Challenge 3: Get TurtleBot to Draw on a Path



Intent:

- Pen retracts and extends w/ command.
- Modified onto TurtleBot 3

Reality:

- Appears, but is white and does not move with TurtleBot

# Happy Little Challenge 4: Simplifying Images

```python
#Code from:
# https://github.com/harshitroy2605/imag-to-sketch/blob/master/1.py

import numpy as np
import imageio
import scipy.ndimage
import cv2

img="BotROS.png"

def grayscale(rgb):
        return np.dot(rgb[...,:3],[0.299,0.587,0.114])

def dodge(front,back):
        result=front*255/(255-back)
        result[result>255]=255
        result[back==255]=255
        return result.astype('uint8')


s=imageio.imread(img)
g=grayscale(s)
i=255-g

b=scipy.ndimage.filters.gaussian_filter(i,sigma=10)
r=dodge(b,g)

cv2.imwrite('1.png',r)
```

- Figuring out how to create an image simple enough for BotROS to recreate.
- Initially thought to create vector images that can be mapped out.
- Textures and colors become an issue.

# What's next?

- Implement Other Colors
- Improve Pathing Algorithm
  - Speed
  - Efficiency
  - Course-correcting
- Increased Complexity Pathing Algorithm
- ???
- Profit.

Questions?