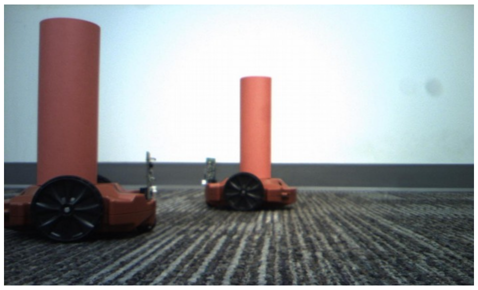
LAB 4: TRACKING WITH PARTICLE FILTERS

Due: Wednesday, April 20th

6pm code/visualizations

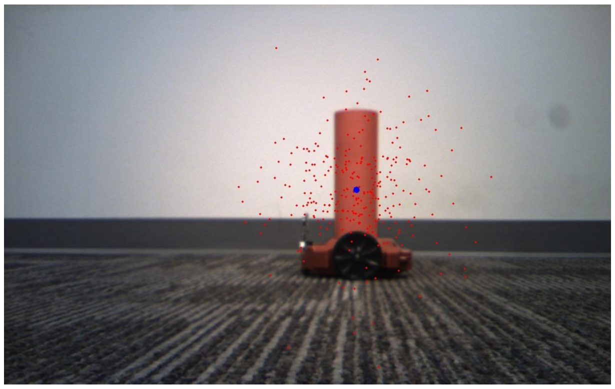
Business is booming at PatrolCorp and clients are demanding more and more robots for surveillance. Unfortunately, more robots has meant more problems. Clients have been complaining to support that their robots keep running into each other and stalling. The planning team has devised a collision avoidance scheme to keep robots from running into each other, assuming the robots know where each other are. It's up to your team to develop a tracker to monitor the motion of nearby robots.

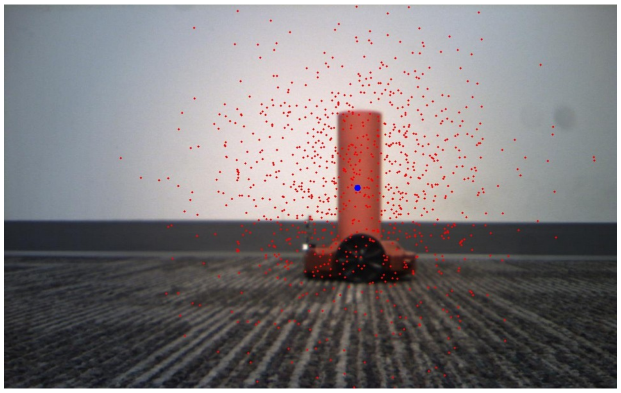


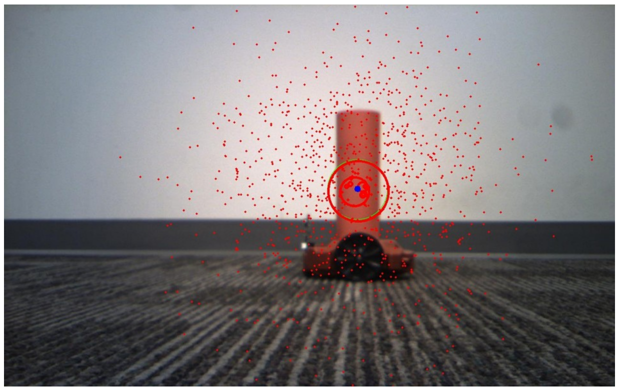
For this project, you will develop a particle filter for tracking the motion of nearby robots. Download `Lab4.py` from Canvas. As part of this lab, you will need to complete the following functions:

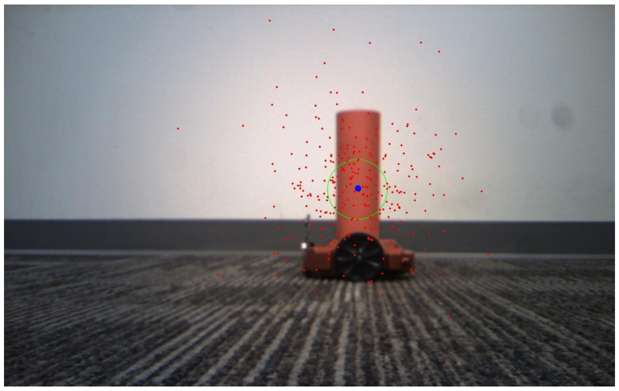
1. `__main__`**:** Complete the function to initialize your particles using a uniform distribution.
2. `detectBlobs(im)`**:** Given an image, this call should should locate the potential location(s) of the red marker on top of the robot. Note than more than one marker may be in the image, or none at all. We recommend using OpenCV's `cv2.SimpleBlobDetector()`
3. `predict(particles, predictSigma)`**:** Implement the prediction step, which is equivalent to a motion update in localization. Since we don't have an explicit motion model for the object we are tracking in this case, model this update using a Gaussian distribution.
4. `update(particles, weights, keypoints)`**:** Implement the sensing update in which you calculate the importance weight of each particle. Remember to renormalize the weights to maintain a probability distribution.
5. `resample(particles, weights)`**:** Implement the resampling step in which you probabilistically resample the particles with replacement. You may also add random particles or use any other algorithmic variants you can think of to improve the performance of your algorithm.

Shown below is an example tracking iteration with particle and blob visualizations provided by the starter code:


1. Current Particles


2. Predicted Forward


3. Updated Weights


4. Resampled Particles

Note:

Your blob detection does not have to be perfect. The point of the particle filter is to improve tracking in the face of unreliable and missing detections. That being said, reliable blob detection is fairly easy to achieve with minimal tuning (we can detect the red marker on the robot in all but 2 frames when it is visible in both image sets).

Evaluation:

Submit the following files on Canvas by 6pm on April 20th.

a. A zip file containing your code. (*Lastname1Firstname1_Lastname2Firstname2_code.zip*)

b. A zip file containing all images generated for ImageSet1 and ImageSet2 (*Lastname1Firstname1_Lastname2Firstname2_images.zip*

   For each image in the image sets, submit the 4 images generated by the starter code (shown above). For example, for image 01 of ImageSet1, the 4 submitted images should be: *ImageSet1_01_1_Current.jpg*, *ImageSet1_01_2_Predicted.jpg*, *ImageSet1_01_3_Reweighted.jpg*, *ImageSet1_01_4_Resampled.jpg*

c. Each partner should complete the peer evaluation form (please complete this even if you a working without a partner).

Grading Rubric

Implement blob detection	30 points
Implement prediction	20 points
Implement update	25 points
Implement resampling	25 points
Peer evaluation form	5 points