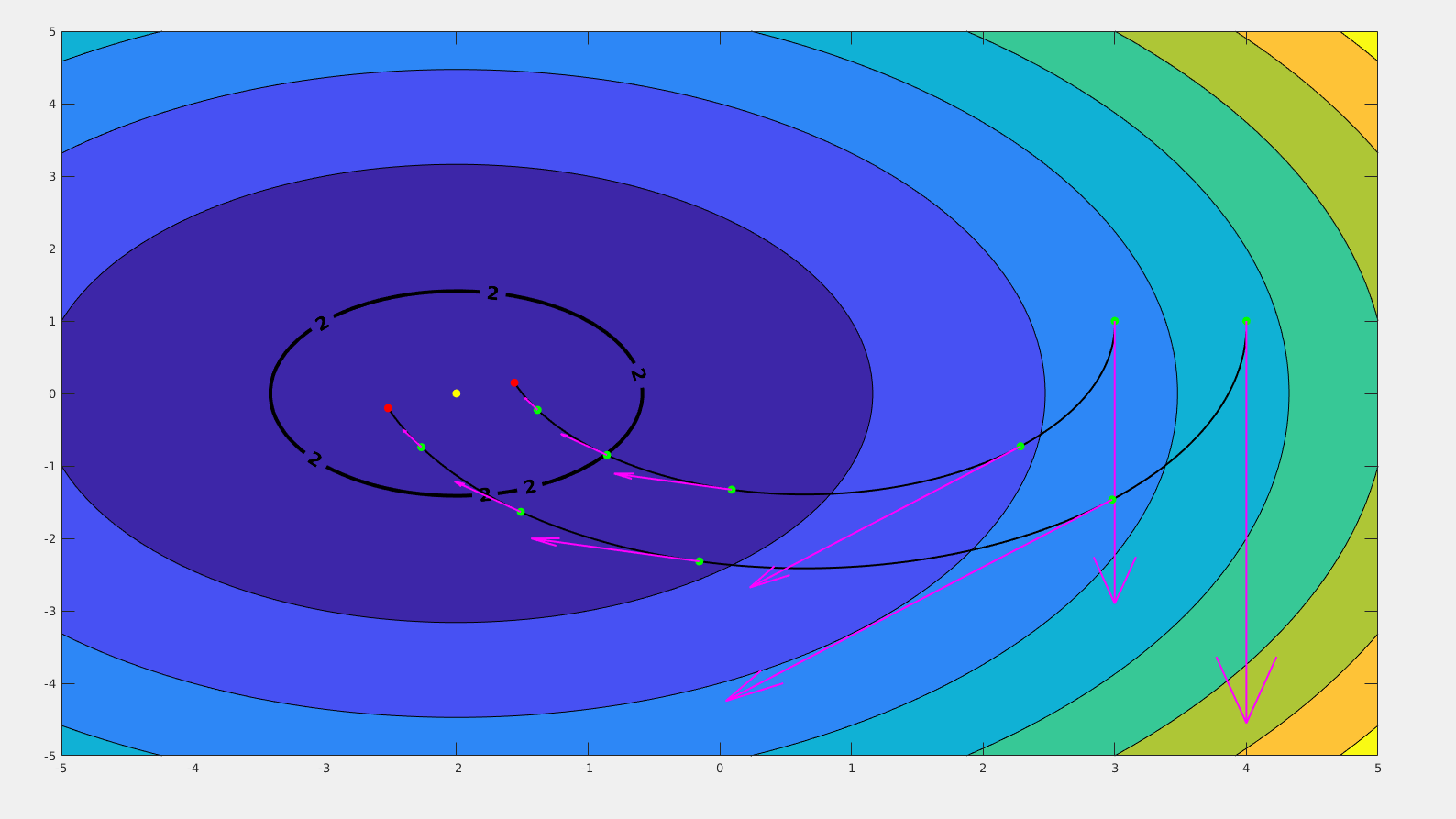
**Problem 1**

(a)

(b)

(i)



(ii) See attachments

(c)

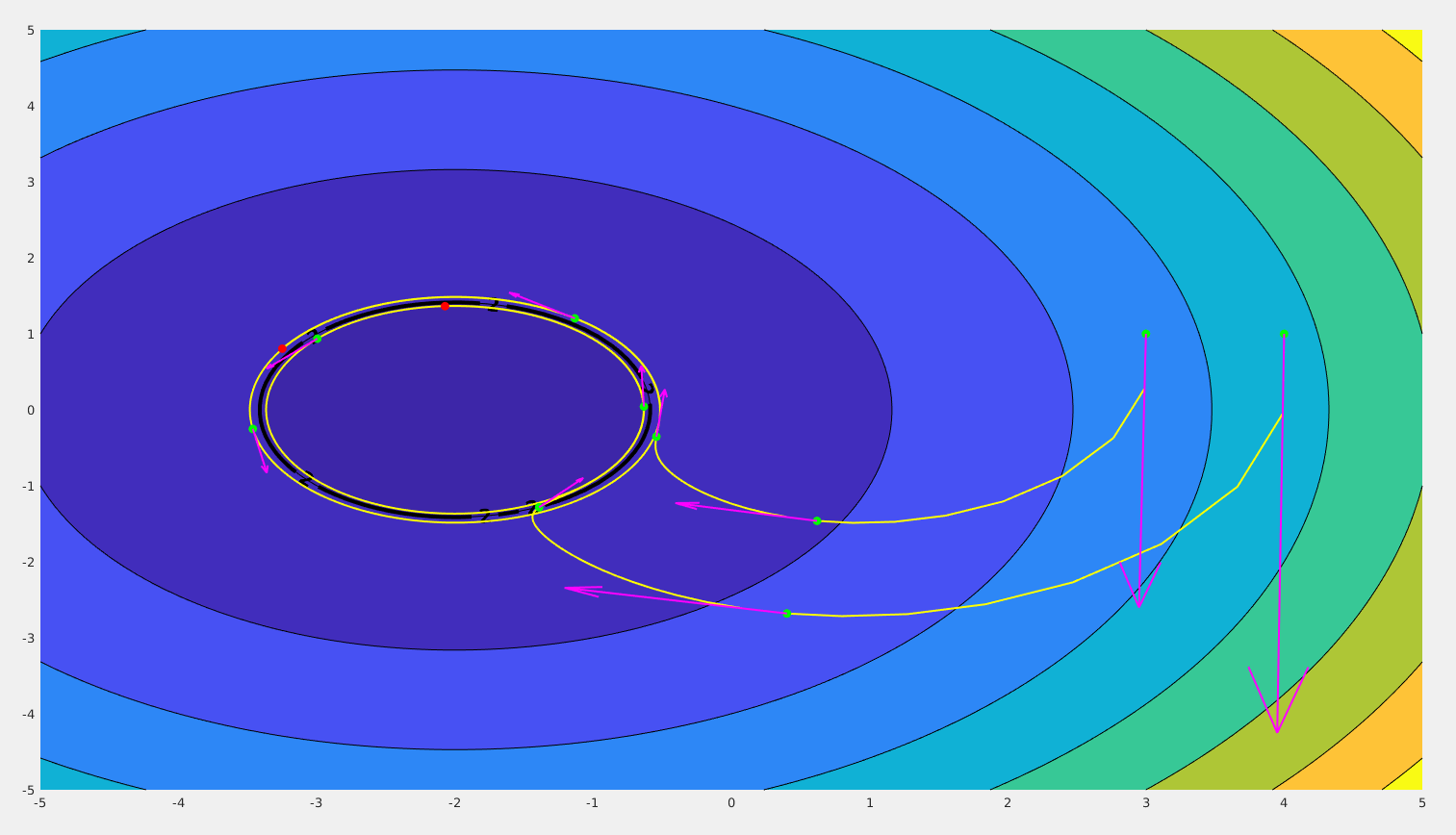
The controller works by treating the two robots similarly to the front tires of a car with differential steering. While the two robots are not at the target, the controller tells the one furthest away to move faster than the other. The end result of that command is the two robots moving in unison along a near circular pattern and pointing more towards the target. This eventually leads to the robots moving towards the target and arriving.

**Problem 2**

**(a)**

**(b)**

**(i)**



(ii) See attachments

**(c)**

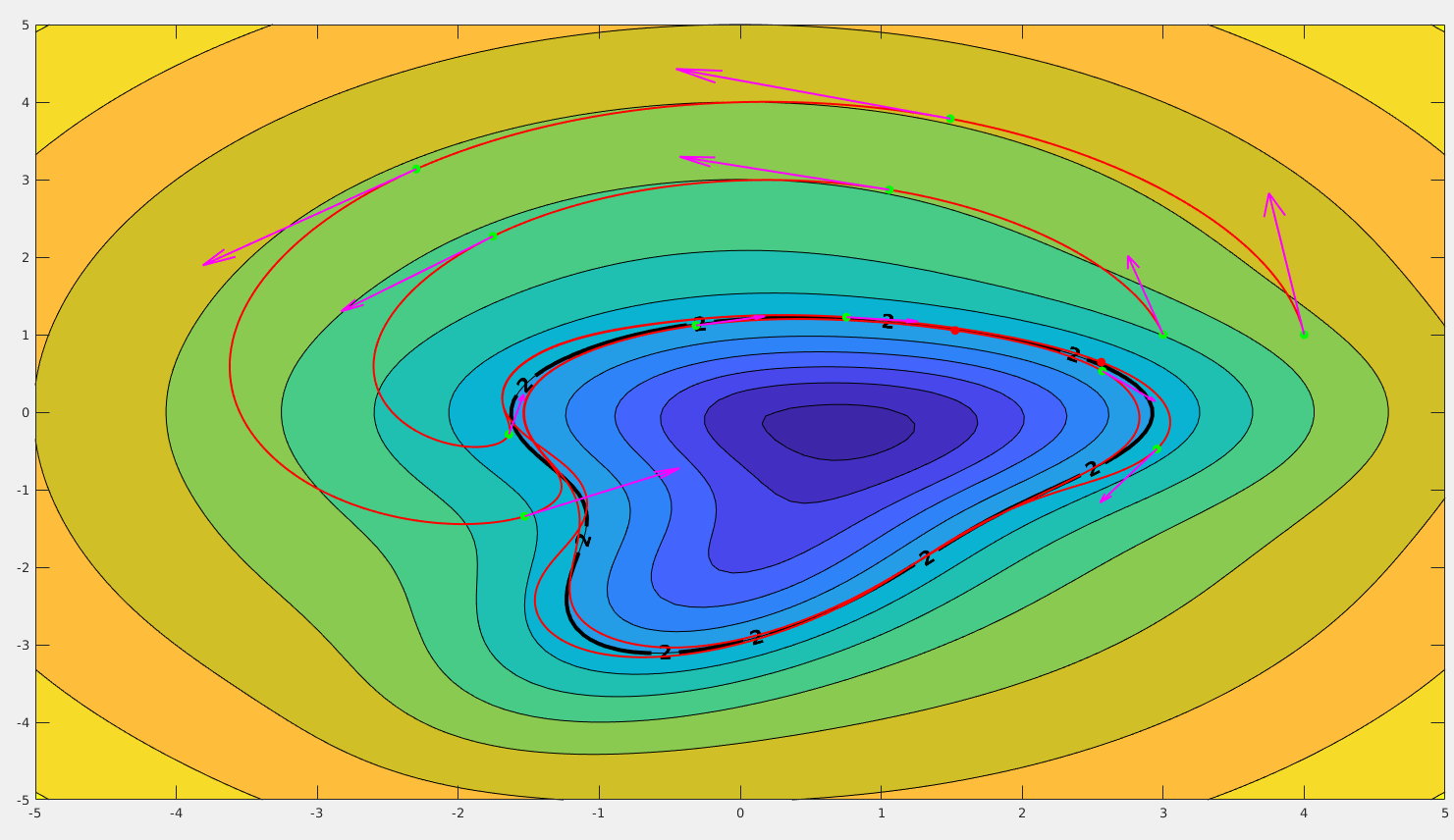
If then the robots would stop moving if they ever actually reach the level curve will cause the tracking to drift slightly during the initial approach, but also ensures that the robots track the level curve and don't get stuck. If becomes larger (closer to ), then the robot will not track the curve as well.

**(d)**

The first part of the control equation drives the robots towards the level curve by moving the robots furthest away at a higher velocity than the closer one thereby causing the robots to "face" the level curve. The second term in the equation is overshadowed by the first at a large distance, but causes the robots to move tangent to the level curve when close and facing the level curve. When moving tangent moves the robots off the level curve, then the first term will move the robots closer proportional to the distance they are away from the curve.

(e)

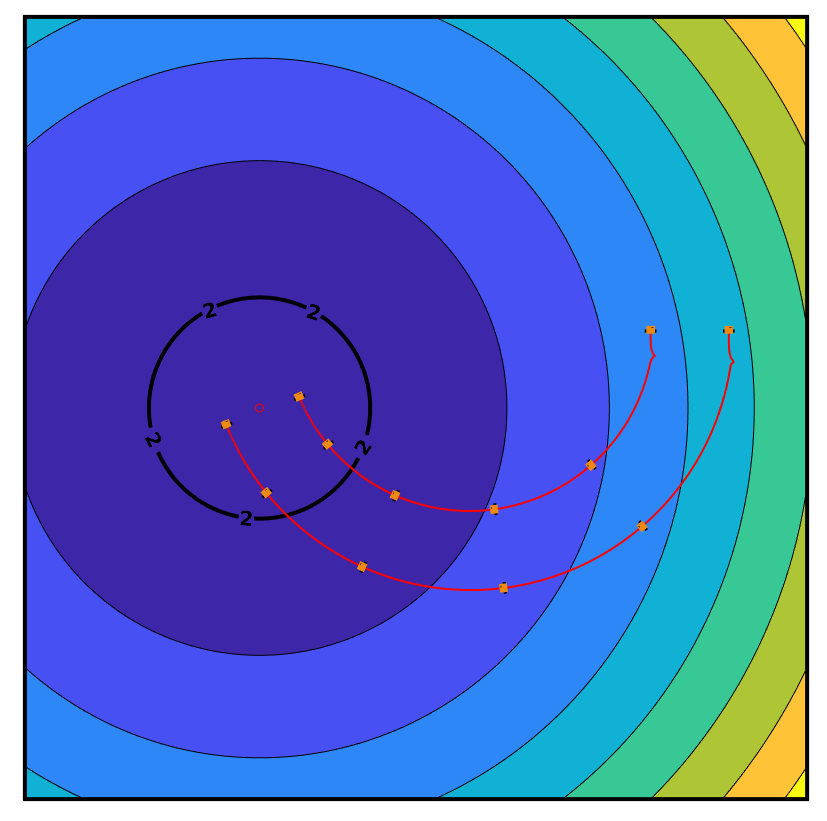
(i)

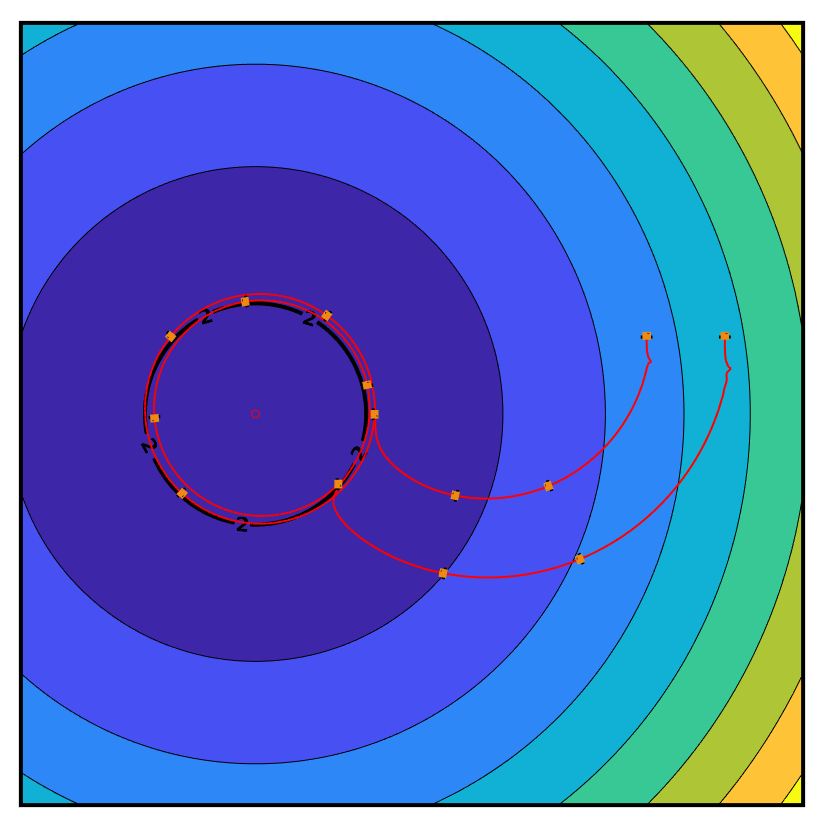


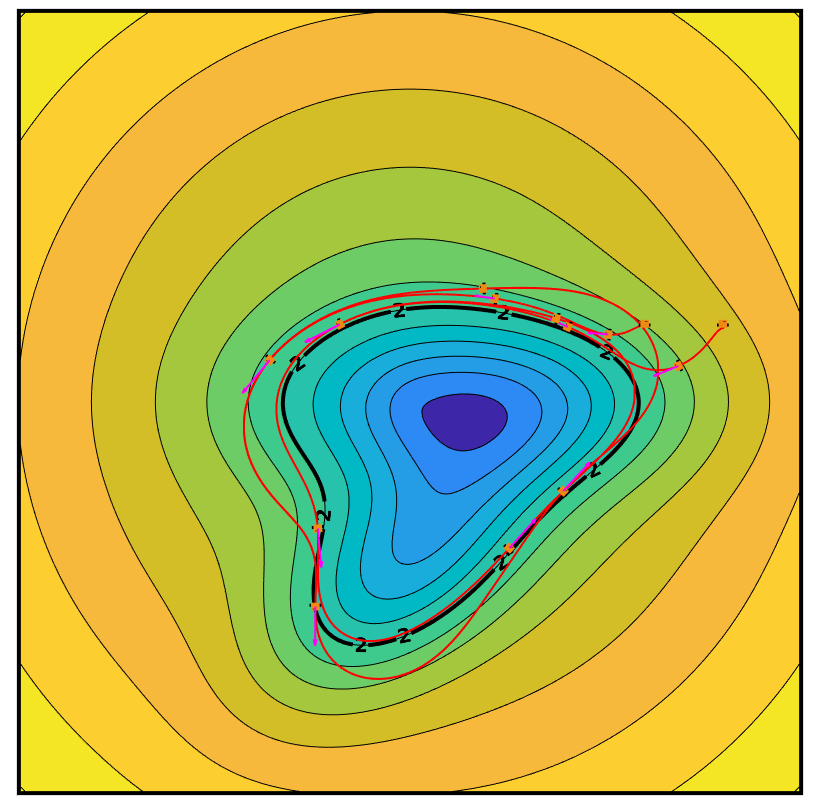
(ii) See attachments

**Problem 3**

Implementing the algorithm on two unicycle robots yielded similar results to problems 1 and 2; however, given the dynamics of the unicycle, the tracking was not as smooth. For instance, when implementing the first velocity controller, there was a brief hiccup in the initial trajectory. Overall, the unicycle model was able to be controlled by these controllers.

Part 1

Part 2b

Part 2

*See attachments for animations*